Departamento de Informática

PROYECTO FIN DE CARRERA

Diseño e Implementación de un Protocolo Seguro de Intercambio de Mensajes con un Dispostivo Seguro

Autor: Ana María Vázquez Pacheco

Tutor: José María Sierra Cámara

Leganés, julio de 2010

Título: ***Diseño e implementación de un protocolo seguro de intercambio de mensajes con un dispositivo seguro***

Autor: Ana María Vázquez Pacheco

Director: José María Sierra Cámara

EL TRIBUNAL

Presidente: Miguel Ángel Ramos

Vocal: Javier Fernández Muñoz

Secretario: Julián Urbano

Realizado el acto de defensa y lectura del Proyecto Fin de Carrera el día 12 de Julio de 2010 en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de

VOCAL

SECRETARIO PRESIDENTE

Agradecimientos

A mis padres Santiago y Albina.
A mis hermanos Mayte y Santi

A todos los miembros del laboratorio “Evalues”
del Parque Científico-Tecnológico de Leganés.

A José María Sierra

Resumen

Debido a restricciones físicas, de seguridad y de coste, las plataformas confiables normalmente son incapaces de interactuar directamente con el usuario. La mayoría de las tarjetas inteligentes dependen de dispositivos de lectura no portables y muy caros para poder mostrar mensajes y recibir datos de los usuarios. En la mayoría de los casos los lectores de tarjetas son proporcionados por la entidad que no depende del usuario. La forma de solucionar este problema pasa por idear un protocolo de seguridad que permita a un usuario intercambiar mensajes con una plataforma segura de tal manera que la integridad y autenticidad de los datos sea incuestionable. Además, debe de ser posible que las acciones requeridas para la realización del protocolo puedan ser ejecutadas por una persona sin ayuda de un dispositivo informático.

Se desarrollará un prototipo que permita realizar firmas digitales de un mensaje con una tarjeta inteligente en entornos no confiables sin que la integridad de la petición de firma se vea afectada por cualquier programa malicioso que se encuentre activo en dicho entorno. El objetivo principal del proyecto se centrará en encontrar una solución que permita que sea el propio usuario el verificador de la integridad de la petición. El dispositivo criptográfico no llevará a cabo ninguna operación sin antes verificarla y obtener una confirmación explicita por parte del usuario de la petición.

El protocolo seguido para realizar la firma digital es el siguiente:

1. **Acceso al sistema.** El usuario introducirá su código PIN en la aplicación del terminal, este enviará la instrucción de verificación de número PIN a la tarjeta inteligente donde se verificará.
2. **Inserción del mensaje.** Si la tarjeta inteligente confirma que el número PIN introducido en el paso 1 es correcto, el terminal pide mediante un mensaje por pantalla que el usuario introduzca el mensaje que desea firmar. El terminal recoge el mensaje y se lo transmite a la tarjeta inteligente.
3. **Petición de CAPTCHA.** Cuando la tarjeta inteligente recibe un mensaje, lo almacena y genera una petición de CAPTCHA para enviar al servidor generador de CAPTCHAS. La petición está compuesta por tres campos:
    - Mensaje a firmar enviado por el usuario + OTP cifrado por la clave pública del servidor.

- Firma digital del mensaje cifrado.
- Identificador de usuario.

4. **Envió de la petición.** La tarjeta transmite al terminal la petición y este se encargará de enviarla hasta el servidor de CAPTCHAS a través de una conexión.
5. **Generación CAPTCHA.** El servidor comprueba que conoce al usuario y que tiene almacenada su clave pública. Comprueba la integridad del mensaje mediante la firma, si es capaz de verificarlo, entonces, descifrado el mensaje y procede a generar un CAPTCHA que contenga el mensaje y el OTP con la clave asociada al id del usuario incluido en la petición.
6. **Retorno de CAPTCHA.** El CAPTCHA se envía al terminal mediante la misma conexión iniciada en el paso 4. El terminal procede a mostrarlo.
7. **Confirmación del mensaje.** Si el mensaje es autentico el usuario introduce la clave OTP en el terminal, que deberá transmitirla hasta la tarjea.
8. **Firmar mensaje.** Cuando la tarjeta inteligente recibe un OTP correcto firma el mensaje que había almacenado en el paso 2 y se la envía junto con el mensaje al terminal.

Palabras clave: protocolo comunicación, mensaje autenticado, dispositivo de seguridad, terminal no confiable, firma digital, criptografía visual, tarjeta inteligente, CAPTCHA.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

INTRODUCCIÓN Y OBJETIVOS

1.1 Introducción

La seguridad de la información, aunque se centra en la protección de todos los activos de los sistemas de información, puede entenderse como la consecución de un conjunto de determinados servicios, de los que pueden extraerse como fundamentales: la confidencialidad, la integridad, la disponibilidad y la autenticación. Los ataques contra ésta última, la autenticación, justifican el trabajo realizado en este proyecto.

Si por autenticación entendemos el proceso de establecer la confidencialidad de las identidades de una entidad, por protocolo de autenticación on-line debe entenderse como el proceso de intercambio de mensajes bien especificado que verifica la posesión de un elemento identificativo físico o lógico (tokens) por parte de una entidad, para remotamente autenticar a la misma. Algunos protocolos de autenticación también generan claves de cifrado para proteger una sesión completa, de forma que los datos transferidos en dicha sesión quedan criptográficamente protegidos. Bajo esta definición, un dispositivo también tiene asociado una identidad y, más allá, podría participar en ciertos protocolos de autenticación de forma autónoma e independiente a la existencia de un usuario.

En lo referido a la autenticación de usuarios, las tarjetas inteligentes o smart cards han desempeñado un papel fundamental en nuestra historia reciente. Los enormes progresos en microelectrónica realizados a principios de los años setenta, hicieron posible integrar en un tamaño considerablemente reducido el almacenamiento de datos junto con un microcontrolador, en un único chip de silicio. El registro de la primera patente de tarjeta

inteligente (año 1974), el descenso en el precio de los circuitos integrados y la aparición de las primeras memorias no volátiles fueron otros factores que enmarcaron la antesala de la considerada como primera generación de estos dispositivos.

Desde el comienzo de su desarrollo, las tarjetas inteligentes proveían un medio ideal para la criptografía. Podían almacenar de forma segura material criptográfico, claves de cifrado y firma, certificados digitales, contraseñas e incluso, con el tiempo, patrones biométricos; al mismo tiempo, estaban capacitadas para ejecutar algoritmos criptográficos, cada vez más potentes. Por otro lado, destacaban por su forma, tamaño y facilidad de manejo, de tal manera que podían ser empleadas en una variedad de contextos por el usuario.

En los últimos años, el manejo masivo de información sensible, ha permitido el desarrollo y una mejora sustancial en las aplicaciones y protocolos de seguridad de la tarjeta inteligente. En un contexto organizacional éstas permiten: acceso seguro y autenticado de usuarios a sistemas o redes, dependencias corporativas, etc., y en todo caso, almacén y procesado de credenciales y material digital de seguridad. Con la inclusión de las tarjetas inteligentes como un elemento más de la denominada Sociedad de la Información, conviene ahondar en las aplicaciones, protocolos o mecanismos en los que se ponen en juego datos personales y, por tanto, susceptibles de ser protegidos haciendo uso esta tecnología. La importancia de la vinculación entre dicha tarjeta inteligente de forma presencial y la autenticación de usuarios puede verse claramente reflejada en una diversidad de escenarios.

Aunque hoy por hoy las tarjetas inteligentes son consideradas dispositivos avanzados, con interesantes recursos criptográficos y computacionales comparado con su reducido tamaño, el diseño de los protocolos en los que interviene, o incluso las aplicaciones en las que se hace uso, a menudo no han ido más allá de considerarlos como unos elementos de soporte en el esquema global. En este sentido, carecen de la autonomía que se le presupone a cualquier otro nodo de red con capacidad de incorporar protocolos de comunicación acorde con el resto del sistema, cada vez más diverso e interoperable.

1.2 Planteamiento del problema y motivación

Los protocolos de seguridad frecuentemente requieren que las entidades implicadas en su ejecución realicen complejas operaciones computacionales. Mientras que los ordenadores, y máquinas en general, están ideados para llevar a cabo estos trabajos, las personas no. Los usuarios humanos están condenados a confiar en que las máquinas realicen correctamente estas operaciones en su nombre [1]. La ubicuidad de los terminales de acceso público es conveniente, pero estas máquinas no actúan bajo los intereses de sus usuarios, sino en el de los programas que están instalados en ellos. Incluso aunque el propietario de un ordenador es confiable, no existen garantías de que los sistemas no se encuentren bajo el control de un programa atacante.

Debido a restricciones físicas, de seguridad y de coste, las plataformas confiables normalmente son incapaces de interactuar directamente con el usuario. La mayoría de las tarjetas inteligentes dependen de dispositivos de lectura no portables y muy caros para poder mostrar mensajes y recibir datos de los usuarios. En la mayoría de los casos los lectores de tarjetas son proporcionados por la entidad que no depende del usuario. En el caso de una

plataforma segura perteneciente a una red de ordenadores, el usuario es claramente dependiente de un terminal local para poder transmitir mensajes a través de la red.

Figura 1. Escenario de uso de una trajeta inteligente

El resultado de esta falta de interacción es que un sistema que hace uso de una plataforma confiable, como es la tarjeta inteligente, puede ser vulnerable a ataques realizados por sistemas no confiables que se encuentran situados entre el usuario y la plataforma confiable. Consideremos el caso de un investigador que se encuentra de viaje con motivo de una conferencia sobre seguridad en las tecnologías de la información que es impartida en una universidad. El investigador necesita enviar un mensaje importante a un compañero que se encuentra en la oficina. Dado que es un mensaje relevante decide enviarlo firmado digitalmente haciendo uso de su tarjeta inteligente y enviarlo por correo electrónico. Para realizar esta operación el investigador utiliza uno de los terminales públicos perteneciente a la red de la universidad y que está provisto de un lector de tarjetas inteligentes. Dado que todas las comunicaciones se realizan a través del terminal, que tiene capacidad para modificar los mensajes en tránsito, el investigador nunca podrá estar seguro de que el mensaje que él ha introducido sea el mismo que el que se ha enviado a la tarjeta inteligente.

Más aún, ¿Cómo puede saber si el mensaje realmente ha sido firmado?, ¿Cómo puede conocer si el mensaje devuelto por la tarjeta inteligente es el mismo que se muestra en la pantalla del terminal? y ¿Cómo puede comprobar que no ha sido enviado más peticiones sin su permiso a la tarjeta inteligente?

Figura 2. Escenario de un ataque sobre una tarjeta inteligente

La forma de solucionar este problema pasa por idear un protocolo de seguridad que permita a un usuario intercambiar mensajes con una plataforma segura de tal manera que la integridad y autenticidad de los datos sea incuestionable. Además, debe de ser posible que las acciones requeridas para la realización del protocolo puedan ser ejecutadas por una persona sin ayuda de un dispositivo informático. [1]

1.3 Objetivos

Después del análisis inicial del estado de la seguridad en las comunicaciones entre personas y dispositivos de seguridad, surgen los objetivos principales de este proyecto. Con el trabajo realizado en este PFC se pretende:

- **Estudio de los protocolo de seguridad utilizados en la autenticación de las comunicaciones entre personas y máquinas.** Se analizarán los protocolos de seguridad para la autenticación de las comunicaciones entre persona y máquina

- **Estudio de las técnicas utilizadas para la generación de retos Basados en el Test de Turing Invertido.** Se estudiarán las distintas técnicas para la generación de CAPTCHAs (Complete Automatic Public Turing test to tell Computers and Humans Apart), así como otro aspectos relativos como son las técnicas de distorsionado de imágenes o la características de usabilidad necesarias para que una persona puede reconocer el contenido de una imagen.

- **Diseño de un mecanismo de seguridad que permita autenticar a una máquina como emisor de un mensaje.** Se ideará un solución que permita el envío de mensajes de forma autenticada e integra desde una máquina a una persona, sin que la persona deba hacer uso de un dispositivo extra para verificar el emisor del mensaje o comprender en contenido del mismo.

- **Definir un protocolo de comunicación que permita a un usuario intercambiar mensajes con una tarjeta inteligente.** Diseño de un protocolo de seguridad que permita al usuario verificar todas las peticiones realizadas a la tarjeta inteligente sin necesidad de utilizar un dispositivo informático.

- **Implementación de un sistema que realice la firma digital de un mensaje siguiendo el protocolo establecido anteriormente.** Desarrollo de un prototipo en el que se apliquen en un escenario real el mecanismo y protocolos diseñados para la comunicación entre personas y plataformas confiables.

1.4 Fases del desarrollo del proyecto

Fase 1. Análisis del problema y estudio de la documentación.

Esta fase comprende la realización de un exhaustivo estudio sobre los protocolos de autenticación de comunicaciones entre personas y máquinas. Estos estudios se centrarán en aquellos mecanismos que, en particular, permitan la inclusión de algún tipo de clave que autentique a una máquina como emisor de un mensaje sin que una persona necesite un dispositivo computacional extra para detectarla.

El estudio anterior deriva en una investigación sobre las técnicas de generación de CAPTCHAs seguros. Además, se estudian otros aspectos relacionados con la implementación como son el tratamiento y distorsión de imágenes y la generación de bases de datos propicias para la selección de entradas aleatorias basadas en técnicas de seguridad.

Fase 2. Recogida de requisitos.

Esta fase del proyecto se centra en la recogida e identificación de requisitos funcionales y no funcionales. Para ello se necesita conocer los servicios y recursos que facilita el sistema, se definen las características que deben cumplir las comunicaciones intercambiadas en el sistema y se idea la integración del mecanismo de autenticación dentro de un protocolo de comunicación. En esta fase se analizarán con especial cuidado los requisitos de seguridad dada la finalidad del sistema y los requisitos de usabilidad debidos a la importancia de que el usuario final sea capaz de comprender los datos de las comunicaciones.

Fase 3. Diseño de solución.

Esta fase se centra en el diseño de un protocolo de comunicación entre una persona y una plataforma confiable, y de forma más específica, en el diseño de un mecanismo de autenticación para mensajes. Para ello, y partiendo de los requisitos generados en la fase anterior, se realiza el diseño del sistema teniendo en cuenta las principales ventajas de aplicación de cada una de las soluciones tomadas y realizando una descripción detallada del protocolo a implementar.

Fase 4. Implementación de la solución.

En la etapa de implementación se procederá a desarrollar el protocolo de seguridad que permite a un usuario intercambiar mensajes con una plataforma segura de tal manera que la integridad y autenticidad de los datos sea incuestionable siguiendo las directrices descritas en la etapa de diseño.

Fase 5. Pruebas.

Una vez culminada la integración, y dentro de esta misma fase, se realizará la evaluación del funcionamiento del sistema completo. Los resultados de esta evaluación ponen de manifiesto la adecuación del sistema a los requisitos especificados al comienzo del proyecto.

Fase 6. Documentación.

La etapa final consiste en la redacción del documento que describa en detalle los resultados obtenidos en cada una de las fases del desarrollo del proyecto.

1.5 Estructura de la memoria

Capítulo 1 – Introducción y objetivos. Descripción global del problema y motivaciones por las que se debe llevar a cabo el proyecto. También se incluye un breve desglose de las fases de desarrollo del proyecto.

Capítulo 2 – Ámbito del proyecto. Descripción de los campos de la tecnología de la información que tienen influencia en el proyecto mediante la introducción de definiciones básicas y el estudio del estado del arte de las técnicas aplicadas para el desarrollo del proyecto.

Capítulo 3 – Especificación de requisitos de usuario. Descripción de las capacidades y restricciones generales del sistema mediante requisitos de usuario, diagramas de caso de uso y diagrama de actividad.

Capítulo 4 – Diseño. Descripción de la solución tomada para el desarrollo del sistema mediante la especificación de requisitos software y la descomposición de componentes.

Capítulo 5 – Implementación. Descripción del código desarrollado para cada uno de los elementos definidos en la descomposición de componentes de la fase de diseño.

Capítulo 6 – Plan de pruebas. Descripción de la metodología de evaluación del sistema junto con la enumeración de las pruebas realizadas para valorar el correcto funcionamiento del sistema y el cumplimiento de los objetivos iniciales.

Capítulo 7 - Conclusiones y futuros desarrollos. Descripción del aprendizaje del alumno a lo largo del desarrollo del proyecto y posibles mejoras a realizar en el desarrollo.

Anexo A. Presupuesto.

Glosario.

Referencias.

Capítulo 2

ÁMBITO DEL PROYECTO

2.1 Introducción

El ámbito donde se desarrolla esta propuesta es el área de la seguridad en las tecnologías de la información y las comunicaciones y, por lo tanto, las amenazas a las que están sujetos los dispositivos criptográficos móviles, como son las tarjetas inteligente, como las plataformas o sistemas (ordenadores personales, teléfonos móviles, terminales bancarios, etc.) que hacen uso habitual de sus capacidades criptográficas tienen gran interés en esta rama de la tecnología de la información.

Las tecnologías y sistemas actuales se diseñan para ser útiles herramientas que colaboran con las personas para que estas alcancen sus intereses, sin embargo, en ocasiones se nos olvida el matiz de que las máquinas no actúan según las exigencias de sus usuarios, sino que se limitan a actuar según los propósitos de las aplicaciones que se ejecutan sobre ellos. Esta sentencia nos permite afirmar que aún siendo el propietario o administrador de una tecnología confiable, no existen garantías de que el sistema no se encuentre comprometido y actúe bajo el control de un programa malicioso, como puede ser un virus, un gusano u otro tipo de malware, con pretensiones de realizar algún tipo de ataque sobre la información accesible desde la plataforma donde se ejecuta.

2.2 Protocolos de identificación

En esta sección se realizará un breve estudio sobre algunos de los protocolos de identificaciones que utilizan técnicas basados en problemas de la inteligencia artificial. La clasificación de estos protocolos se ha basado en el tipo de entidad que se desea autenticar.

2.2.1 Autenticación de una persona ante una máquina

Sea un sistema de autenticación (P,V) se denomina (α, β, τ) - ejecutable por una persona, si al menos un (1 - α) de la población pueden realizar el proceso P de autenticación de forma satisfactoria en un tiempo inferior τ con una probabilidad superior a (1 - β). [4]

Los protocolos de identificación de una persona frente a una máquina, más conocidos como Protocolo de Autenticación de Humanos ó HAP (Human Autentication Protocol), son los protocolos más comunes en el ámbito de la tecnología de la información, llegando incluso a eclipsar al resto de escenarios.

A continuación se muestran algunos de los protocolos de seguridad diseñados para la identificación de usuarios. Existen mucho más de los que se exponen y con características muy dispares. Los aquí descritos muestran una pequeña pincelada de la evolución de los mecanismo que se han basado en la características humanas para realizar la autenticación de los usuarios.

Protocolo Hopper-Blum (HB)

En 2001, Hopper & Blum [4] idearon un protocolo de seguridad para la autenticación de personas, también conocido como el protocolo HB.

Su funcionamiento es sencillo y se basa en un esquema resto-respuesta. El usuario y el sistema comparten un clave secreta compuesta por K-bits. Cuando el usuario desea autenticarse ante el sistema recibe un reto formado por una secuencia aleatoria de 1's y 0's. El usuario debe responder con el resultado de realizar la multiplicación binaria del reto con la clave compartida. El sistema realizará la misma operación y si la paridad de bits de la respuesta de usuario es correcta aceptará la autenticación. Si el reto está constituido por una única operación, un atacante que no conozca la contraseña tendrá un 50% de posibilidades de ser aceptado como un usuario válido. A medida que aumenta el número de operaciones a realizar para autenticarse la probabilidad de que un atacante sea capaz de acertar introduciendo valores aleatorios decrece considerablemente. Sin embargo, un proceso espía que capture k pares de mensajes reto-respuesta válidos entre el usuario y el sistema podrá conocer el valor de la clave. Para evitar este ataque, el usuario podrá introducir bits de ruido en la respuesta. Pero en general, el protocolo HB es muy limitado ya que tan sólo provee seguridad ante atacante pasivos siendo vulnerables a ataques activos.

Jules & Weis ampliaron el protocolo HB para evitar las vulnerabilidades ante atacantes activos, esta extensión se denominó protocolo HB+. En esta nueva versión el secreto compartido constaba de dos vectores de bits así como el reto, el usuario debía realizar dos multiplicaciones, el primer vector de la clave con el primer vector del resto y el segundo vector de la clave con el segundo vector del reto, y

enviarle al sistema el vector resultante de realizar un XOR con ambos resultado, el sistema continua comprobando la paridad de la respuesta para validar la autenticación. Esto hace que el protocolo sea aún más complejo de realizar correctamente por una persona

Esquema seguro de autenticación cognitiva contra procesos espías

Weinshall [2] introduce un esquema de autenticación que solamente depende de la funciones cognitivas de los humanos. En este protocolo el usuario debe acordar previamente un conjunto de imágenes con el proceso de verificación a modo de clave compartida. En la autenticación el sistema envía al usuario una matriz de imágenes, este tendrá que recorrer desde la esquina superior izquierda hasta la inferior derecha la maya de imágenes y enviar al verificador el número asociado a la posición de cada una de las imágenes que componen la clave. El protocolo es seguro frente a ataques de fuerza bruta debido a la relación entre el número de imágenes para seleccionar y el tamaño de la clave del usuario.

En la Figura 3 se muestra un ejemplo del protocolo ideado por Weisgall, la línea dorada denota el camino de imágenes que el usuario debe reconocer como clave.

Figura 3. Ejemplo protocolo de autenticación cognitiva

Método de entrada de PIN contra "shoulder surfing"

El *shoulder sufing* es el término que se ha asociado a aquellas situaciones en las que al introducir una clave o PIN en un lugar público pueda haber alguien, ya sea físicamente o mediante cámaras, espiando las entradas del usuario y descubra la clave de identificación del usuario sin que este se dé cuenta.

Roth, Richther, y Freidinger introducen un mecanismo muy simple para realizar una autenticación segura mediante entrada de PIN. El protocolo asume que el atacante tiene una capacidad de memoria limitada y no puede almacenar el conjunto de interacciones en su cabeza.

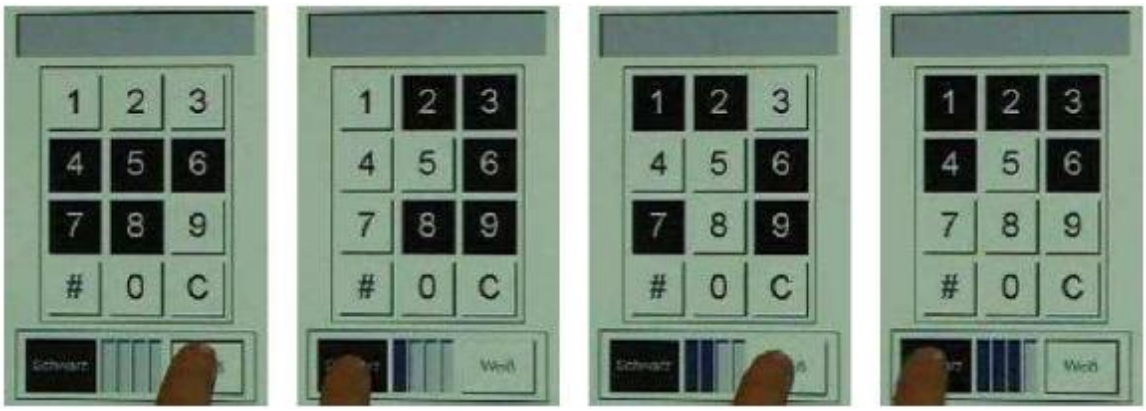

Figura 4. Ejemplo de protocolo contra shoulder surfing

El sistema para cada uno de los dígitos que componen el PIN envía al usuario cuatro imágenes. Las imágenes contienen un panel con diez dígitos, cinco de ellos en blanco y los otros cinco en negro. El usuario deberá marcar como blanco o como negro dependiendo del grupo en el que se encuentre el digito del PIN que está enviando. Un ejemplo de cómo se introduce en número 3 se muestra en la figura $X, 3 = \{1, 2, 3, 9, 0\} \cap \{2, 3, 6, 8, 9\} \cap \{3, 4, 5, 8, 9\} \cap \{1, 2, 3, 4, 6\}$.

2.2.2 Autenticación de una máquina ante una persona

Existen muy pocas implementaciones en las que sea una persona la responsable de autenticar la identidad de un mensaje generado por una máquina.

Autenticación e identificación visual

En la criptografía humano–máquina, más conocida como Human Computer Cryptography o HCC, una persona, normalmente con la colaboración de un dispositivo externo, ejecuta operaciones criptográficas para codificar un mensaje. El algoritmo de encriptado deberá de cumplir las mismas propiedades de seguridad que aparecen en los mensajes encriptados mediante la criptografía ordinaria realizada por máquinas.[2]

No existen muchos trabajos sobre esta área de la criptografía. Aunque las soluciones están diseñadas para ser HCC, en ellas no toman partido de las características que poseen en las personas y simplemente toman roles similares a las máquinas lo que les obliga en la mayoría de los caso a hacer uso de dispositivos externos donde se ejecutan los algoritmos.

El ejemplo más significativo de la criptografía hombre-máquina es la criptografía visual. Existen dos protocolos que implementan esta técnica.

El primer protocolo divide el área de la imagen que el servidor envía al usuario en dos partes mediante una transparencia o máscara: la zona de contenido y la zona en negro. Cuando el servidor

desea transmitir un mensaje al usuario escribe el texto del mensaje en la zona de contenido y envía la imagen a través de la cámara. El usuario recibe el mensaje y coloca su propia trasparencia sobe el mensaje para verificar que la zona negra está completamente negra y que hay un mensaje en la zona de contenido. Si cualquiera de estas propiedades falla, entonces, el mensaje es rechazado. Un usuario con los ojos entrenados puede detectar una modificación con una probabilidad del 50%.

El segundo protocolo modifica ligeramente el protocolo anterior para hacerlo más seguro ante ataques. En lugar de dividir la transparencia en dos regiones dibuja un rectángulo en ella. El sistema envía el mensaje escribiéndolo dentro del área delimitada por el rectángulo. Al recibir el mensaje el usuario puede verificar que el mensaje se encuentra dentro del rectángulo y que el resto de la imagen es negra.

A user-friendly approach to human authentication of messages

Figura 5. Ejemplo de protocolo de autenticación mediante CAPTCHA 3D

La propuesta descrita por King y Santos en este artículo es uno de los más interesantes y novedosos. El mecanismo diseñado se basa en la utilización de un CAPTCHA para que el usuario pueda autenticar un mensaje. En el esquema el sistema desea enviar un mensaje al usuario, para ellos, se codifica el mensaje transformándolo en un texto en tres dimensiones y junto con otros objetos que hacen la función de clave se genera una imagen en 3D. El usuario puede verificar que los objetos que aparecen en la imagen corresponden con los que previamente habían acordado con el sistema y puede leer el mensaje sin ningún problema. El sistema puede incluir algún objeto o elementos estará como paredes para que un atacante sea incapaz de determinar cuáles son los objetos que hacen la función de clave compartida entre el usuario y el sistema confiable.[1]

2.3 CAPTCHAs

Se denomina CAPTCHA a un reto generado de forma automática que puede ser resuelto por la mayoría de las personas pero, que basándose en la tecnología actual, una máquina no es capaz de resolver. CAPTCHA es el acrónimo de su definición en inglés: Complete Automatic Public Turing test to tell Computers and Humans Apart.

Los antecedentes presentados a continuación se centran en la evolución de las técnicas para la generación de retos tipo CAPTCHA, como la Inteligencia Artificial (IA) ha evolucionado para poder superarlos y en qué situación nos encontramos en la actualidad. En este recorrido a lo largo de algoritmos de construcción de retos nos centraremos en los CAPTCHAs basados en texto y en los basados en imagen, son estas dos técnicas las de mayor aplicación en la construcción de un reto para la autenticación mediante criptografía visual.

2.3.1 CAPTCHAs basados en texto

Existe una gran variedad de CAPTCHAs, los más conocidos, y usados hoy en día, son los basados en texto que explotan la dificultad que las máquinas tienen para poder localizar e identificar una palabra incluida en una imagen.

Por lo general, los CAPTCHAs basados en texto son creados de la siguiente manera:

1. Se elige una palabra de un diccionario predefinido.
2. Se le aplica un formato a la palabra y se convierte en una imagen; algunas técnicas incluyen un fondo a la imagen.
3. Se finaliza la composición degradándola con la aplicación de distintos tipos de distorsiones.

Los métodos de generación de CAPTCHAs se diferencian entre ellos por los algoritmos de elección de palabras/diccionarios, el formato que aplican a los caracteres y las degradaciones realizadas sobre las imágenes.

A continuación se detallan las técnicas de generación de CAPTCHAs basados en texto más populares.

Gimpy (M. Blum et al, 2000)

Gimpy es la primera implementación con usos comerciales que apareció en la Web. Es el más sencillo de los CAPTCHAs, en el que desde un primer momento se intenta evitar los ataques producidos por los OCR (Optical Character Recognition) aplicando degradaciones a la imagen utilizando la herramienta Gimpy.

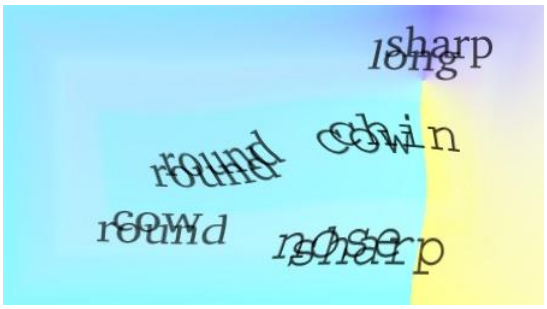

Figura 6. Ejemplo CAPTCHA Gimpy

La construcción de un CAPTCHA de tipo Gimpy es simple:

1. Sobre un diccionario de palabras dado se eligen al azar una de ellas.
2. Esta se transforma en imagen aplicando una fuente que será idéntica para todos los caracteres y un fondo o background.
3. La imagen resultante se degrada aplicando las técnicas facilitadas por el editor de imágenes Gimpy.

Las distorsiones más comunes son la inclusión de fondos con degradaciones y líneas tipo parrilla, aplicación de deformaciones no lineales, difuminación de la imagen total y la inclusión de pixeles de ruido de fondo.

El método Gimpy original consiste en un reto formado por siete palabras, que pueden repetirse, repartidas por la imagen de forma aleatoria, normalmente superpuestas dos a dos; el usuario debía reconocer tres de ella para superarlo. Como se puede observar en la Figura 6 identificar alguna de las palabras que aparecen en la imagen representa un gran reto, no sólo para una máquina, sino también para cualquier humano (sin contar el tiempo que se debe destinar para resolverlos). Dado que esta técnica no cumplía los requisitos mínimos de usabilidad y que la seguridad que aportaba este diseño no era proporcional al problema planteado se evoluciono hacia EZ-Gimpy. Esta nueva aproximación se generaba de igual manera que Gimpy pero se simplificaba al aparecer una única palabra. EZ-Gimpy fue uno de los CAPTCHAs más usuales en la red a principios de la década, y fue usado en sitios web tan relevantes como Yahoo!

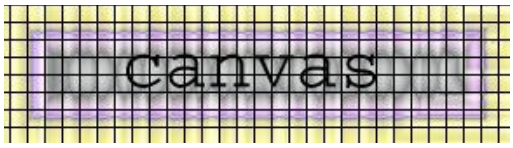

Figura 7. Ejemplo CAPTCHA EZ-Gimpy

Los CAPTCHAs generados siguiendo está técnica pronto fueron rotos, uno de sus puntos más débiles es su limitado diccionario constituido por 860 palabras en inglés, donde no se incluían palabras derivadas como son plurales o conjugaciones verbales (posteriormente se intento solucionar este problema eliminando las limitaciones del diccionario). Pero esta no ha sido la única vulnerabilidad encontrada en esta técnica, la poca variedad de fuentes aplicadas a los caracteres o las distorsiones aplicadas a las imágenes han hecho de este método un objetivo superado por la evolución de las técnicas de inteligencia artificial.

Pessimal Print (H. Baird et al, 2000)

La tecnología de los Pessimal Print CAPTCHAs se basa en los defectos que aparecen en los textos cuando se realizan múltiples copias, escaneos o impresiones por imprentas antiguas. Se trata de simular estos errores de impresión en la palabra-reto del CAPTCHA para que su interpretación sea más costosa aunque para una persona que este acostumbrada a leer documentación con estas características no supone un gran esfuerzo. [7]

Algunas de las distorsiones que aplican, y que caracterizan a esta técnica, son: la aparición de letras; el desdibujamiento de los contornos de los elementos que generan márgenes discontinuos; la fragmentación de caracteres debido a segmentos muy finos o directamente debido a la desaparición de ciertos fragmentos del carácter generando así componentes inconexos; la aparición de pequeñas manchas que no corresponden a ningún elemento del texto lo que genera imágenes con ruido; y la utilización de fuentes de letras condensadas y/o cursivas. Todas estas modificaciones sobre un texto suponen un reto para la inteligencia artificial y hacen más compleja la identificación de los caracteres para los sistemas de reconocimiento de texto.

Figura 8. Ejemplo CAPTCHA Pessimal Print

Pero esta técnica también conlleva algunas restricciones ya que requiere que las personas que se enfrentan al reto conozcan el alfabeto Latino, estén familiarizadas con la lengua inglesa y tengan experiencia en la lectura de esta lengua, ya que la fragmentación de los caracteres puede llevar a equívocos. Este es el caso de la Figura 8, donde una persona sin conocimientos de lengua inglesa podría responder al reto con “answcr” tal y como aparece en la imagen en lugar de con “answer” que sería la correcta. Estas restricciones por lo tanto penalizan a niños o personas sin grandes conocimientos del inglés. Otra de los precariedades encontradas en esta técnica es que las palabras se acotan a una longitud entre 5 y 8 caracteres (lo que facilita la realización de ataques de fuerza bruta) y no se incluyen derivaciones, estas restricciones reducen en gran medida el rango de palabras a utilizar y hacen a esta técnica vulnerable frente ataques léxicos.

Dado que es una técnica con 10 años de edad, se han realizado múltiples estudios que sobre la robustez de la imagen generada por Pessimal Print. Los resultados obtenidos demuestran que cuando el atacante conoce de antemano la fuente de letra utilizada probabilidad de éxito de este es del 40%, además, está técnica es también vulnerable a métodos de restauración de imágenes usados por los OCRs actuales.

Una versión más moderna de esta técnica (incluye algunas características de Baffle Test, explicadas más adelante) es utilizada por reCaptcha [15] (dirigido por Luis von Ahn, cooperador de M. Blum). Sistema líder en este campo, proporciona CAPTCHAS gratuitos vía web, su modelo de negocio se basa en aprovechar las respuestas a los retos para transcribir documentos a formato digital.

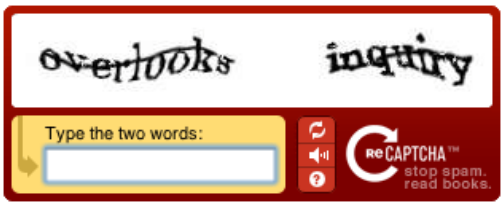

Figura 9. Ejemplo reCAPTCHA

Siguiendo la idea de Pessimal Print CAPTCHA también aparecen los Handwritter CAPTCHA, como su nombre indica en lugar de palabras de imprenta usa imágenes de palabras escritas a mano. Esta técnica no ha llegado a tener gran implantación ya que persisten los problemas de seguridad encontrado en la técnica de Pessimal Print a los que se añade la baja usabilidad de los mismos.

MSN CAPTCHAS (Microsoft, 2002)

Implementación propia de MSN para la generación de CAPTCHAS. Este esquema se utilizó en muchos de los servicios on-line de Microsoft, como son Hotmail, MSN y Windows Live. Fue muy popular hasta que fue roto en el 2006 y sus deficiencias fueron divulgadas en el ámbito científico en el año 2007 con la publicación del artículo "A Low-cost Attack on a Microsoft CAPTCHA" [8]

No se conoce exactamente el algoritmo de generación del CAPTCHA, ya que Microsoft protegió sus implementaciones (con el paso de los años fue incluyendo mejoras tanto de seguridad como de usabilidad) con varias patentes para proteger su tecnología, pero se puede comentar fácilmente las características generales de los retos que proponían a partir de su observación y estudio.

El esquema de estos CAPTCHAS se basa en un reto formado por ocho caracteres, en los que se incluye tanto dígitos como letras, estas últimas siempre en mayúsculas, con una fuente de tipografía serif y color azul oscuro junto con un fondo de imagen liso de color gris claro para asegurar una lectura fácil. Se aplican distorsiones de curvatura sobre el texto tanto a nivel local, en forma de pequeñas olas y las deformaciones elásticas a lo largo de los píxeles del carácter, como a nivel global con deformaciones elásticas de la palabra. Estas deformaciones tratan de frustrar las técnicas en las que se basan los algoritmos detección, como son, el espesor de un carácter o si éste tiene características serif para identificarlo o la aplicación de platillas coincidentes para la detección de fuentes de letra. Además, incrementa la seguridad añadiendo de forma aleatoria arcos de diferentes espesores para generar ruido en la imagen y evitar así los ataques por segmentación.

Figura 10. Ejemplo MSN CAPTCHA

Lo que hace especial el diseño de MSN es que, a diferencia de muchos otros esquemas, aplican texturas de fondo y mallas de colores por encima de la imagen o del fondo a modo de desorden para aumentar la robustez; se aplican arcos de diferentes espesores distribuidos al azar que, además de ruido, también son buenos candidatos como caracteres falsos, y por lo tanto, se esperaba que el diseño proporcionara resistencia a la segmentación. Sin embargo, se ha comprobado que la inclusión de estos arcos de ruido no es suficiente, ya que pueden ser fácilmente localizados una vez eliminados no es muy

costoso identificar la localización de los símbolos en el orden adecuado conociendo que la palabra está compuesta por 8 caracteres.

Baffle Text (H. Baird et al, 2003)

La técnica Baffle Text se basa en los principios de percepción de Gestalt[1] para generar un reto reconocible por un humano y no para una máquina. Estos principios enuncian una serie de leyes sobre el comportamiento de la mente humana, entre ellas, la habilidad que poseen las personas para reconocer formas e imágenes aunque estas se encuentren incompletas o fragmentadas.

El algoritmo de Baffle Text permite construir CAPTCHAs de texto. En primer lugar, se genera una palabra pronunciable, ya que así, un usuario humano tendrá mayor facilidad para responder al reto, pero sin que esta pertenezca al diccionario, para evitar que se produzcan ataques basados en el léxico, para ello, se hace uso de un generador fonético basado en el modelo de Markov. Una vez conseguida la palabra, a la que se le aplica una fuente, que elegida de forma aleatoria entre de un amplio rango de ellas, y se genera una imagen con un fondo simple que no contenga degradaciones. El sistema generará una segunda imagen formada por un conjunto de círculos, cuadrados y/o elipses que se superpondrá a modo de máscara sobre la imagen de la palabra aplicando una operación de suma, substracción o diferencia de contrastes de forma aleatoria entre las dos imágenes.

Casi todas las técnicas actuales para la restauración de imágenes y reconocimiento de caracteres se enfocan en la eliminación de los efectos de desenfoque, de umbral, y localización de pixeles de ruido aleatorio (por ejemplo, los que aparecen aplicando la técnica de Pessimal Print). Sin embargo, hay otros tipos de degradación, como la oclusión o la interferencia de las formas al azar, que borrar partes de la imagen. Los métodos de restauración de la imagen no pueden sustituir las partes que faltan, sin conocimiento previo de las formas de oclusión. A diferencia de las máquinas, los seres humanos son muy buenos en el reconocimiento de una forma completa o imagen a pesar de la información incompleta, dispersa, o fragmentaria.

Figura 11. Ejemplo Bafflet Text

Aunque la metodología utilizada en la generación de CAPTCHAs mediante Baffle Text resulta muy robusta frente a ataques de segmentación de caracteres, su baja usabilidad supone un grave problema en la implantación.

Scatter Type (H. Baird et al, 2005)

Scatter Type es una técnica aplicada en la construcción de CAPTCHAs, basada en lectura, que se diseño para resistir ataques por segmentación de caracteres. Los retos se generan de forma psudoaleatoria sintetizando una cadena de caracteres junto con un formato de letra en una imagen. Dentro de cada imagen, los caracteres de la imagen son fragmentados usando cortes horizontales y verticales, cada uno de los fragmentos generados se dispersan desplazándolos por los ejes de coordenadas x e y. La técnica de dispersión de caracteres se ha diseñado de forma rigurosa para frustrar

[1] Psicología Gestalt. Corriente de la psicología moderna que se centra en el estudio del aprendizaje, la memoria, el pensamiento, la personalidad y motivación humanas.

los ataques por segmentación de caracteres mediante cualquiera de las técnicas que se conocen en la actualidad.

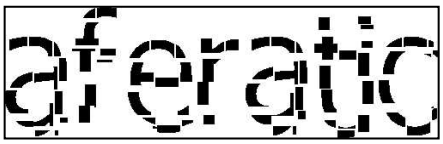

Figura 12. Ejemplo Scatter Type

Como en la técnica de BufferText, para evitar ataques al diccionario, el texto del reto será generado mediante un el modelo de Markov que provee de palabras fonéticamente pronunciables pero que no pertenecen a ningún diccionario conocido. A diferencia de las técnicas estudiadas previamente, Scatter Type no aplica complejas modificaciones sobre las imágenes favoreciendo la usabilidad, siendo los textos altamente reconocibles para las personas aunque el nivel de distorsión aplicado sea muy alto. Nuevamente se han establecido las premisas de Gestalt para el diseño del algoritmo. A pesar de que los recientes esfuerzos para automatizar las habilidades perceptivas de los humanos han supuesto un progreso en el campo, los métodos más conocidos todavía no constituyen una amenaza para ScatterType. Los datos experimentales muestran también que la calificación subjetiva de dificultad está fuertemente relacionada con la ilegibilidad. Además, a medida que se explora el espacio de diseño ScatterType (elección de tipografías, 'palabras', corte de posicionamiento y desplazamientos) con el objetivo de localizar a los rangos en que los retos ScatterType estos siguen siendo cómodamente legibles para casi todas las personas pero se oponen firmemente a los métodos de visión para la segmentación automática en caracteres.

2.3.2 CAPTCHAs basados en imágenes

A diferencia de las técnicas de generación de CAPTCHAs basados en texto, donde el reto que se propone siempre implica averiguar la cadena de caracteres de la imagen, los retos propuestos para resolver los CAPTCHAS de imágenes van desde 'clickear' alguna zona especifica de la imagen; a identificar una serie en las imágenes o incluso formar cadenas de caracteres con las iniciales de los objetos representados por las imágenes que componen el reto.

A continuación se detallan las técnicas más populares de generación de CAPTCHAs basados en imágenes.

Bongo (L. von Ahn M. Blum et al, 2001)

La técnica del CAPTCHA Bongo fue uno de los primeros proyectos del Instituto de Ciencias Computacionales de la Universidad Carnegie Mellon University, cuyo grupo de Inteligencia Artificial, en la actualidad, es puntero en la materia y las tecnologías aplicadas en ella. El reto propuesto en esta técnica se inspira en el conjunto de puzles de reconocimiento de patrones visuales creados por M.M. Bongard[2] en 1967. La aplicación de CAPTCHAs Bongo proporciona dos conjuntos de imágenes: las imágenes de la izquierda pertenecen a una categoría y los de la derecha a otra. El reto consiste en

[2] Bongard, M. M. (1970). Pattern Recognition. Rochelle Park, N.J.: Hayden Book Co., Spartan Books. (Publicación original: Проблема Узнавания, Nauka Press, Moscow, 1967)

determinar a cuál de los dos grupos pertenecen una serie de imágenes sin clasificar (izquierda o derecha).

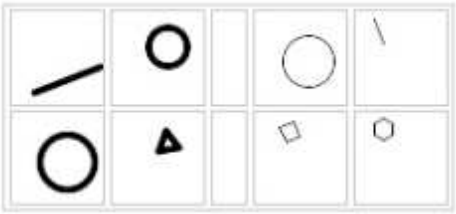

Figura 13. Ejemplo Bongo CAPTCHA

Bongo sólo necesita que se solucionen correctamente cuatro imágenes para superar el reto lo que implica que un ataque con respuesta aleatorias tiene un acierto de alrededor del 6%. Por lo tanto, una máquina sólo tendría que probar, como máximo, 17 veces antes de superar el reto. Se han barajado varias posibilidades para añadir dificultad al reto como aumentar el número de imágenes para clasificar o incrementar las categorías, pero se ha comprobado que estas soluciones tan sólo incrementan las dificultades para las personas y no suponen una mejora de la seguridad ya que para una máquina realizar 1000 iteraciones más no supone un gran esfuerzo.

PIX (L. von Ahn et al, 2003)

Los CAPTCHAs PIX basan la generación de retos en la capacidad que las personas tienen para asociar imágenes con ideas u objetos, como puede ser una mesa, un árbol o un niño. Dado que a una misma imagen pueden asociarse distintos conceptos la técnica desarrollada por L. von Ahn genera un reto que contiene cuatro imágenes distintas sobre una misma idea y el usuario deberá o bien escribir el nombre de la idea mostrada o bien elegir la palabra que mejor describe las imágenes de una lista de opciones lo suficientemente larga para minimizar los aciertos por azar.

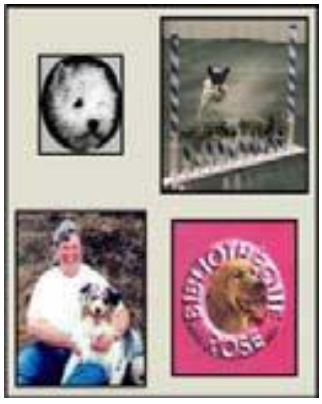

Figura 14. Ejemplo PIX CAPTCHA

PIX basa su potencia en una base de datos que contiene imágenes etiquetadas por conceptos. Dado que las imágenes no se someten a ninguna degradación o transformación la seguridad se basa 100% en la base de datos, una colección de imágenes pequeña o la publicación de esta hacen que un sistema de generación de CAPTCHAs PIX sea muy vulnerable a ataques.

IMAGINATION (R. Datta et al, 2005 - 2009)

IMAGINATION es un sistema de generación de CAPTCHAs basados en imágenes diseñado para ser robusto frente a ataques automatizados ofreciendo a su vez una interfaz amigable para los usuarios. El reto consta de dos pasos que permite reconocer con claridad y rapidez a los humanos, ya que, para una máquina la resolución de los problemas expuestos resulta muy compleja.

Figura 15. Ejemplo de la fase de 'click' en el CAPTCHA IMAGINATION

El primer reto del CAPTCHA consiste en una imagen compuesta, es decir, un mural de imágenes, para superarlo el usuario deberá localizar uno de los límites entre las subimágenes por las que está compuesto y hacer click sobre él. El algoritmo escoge aleatoriamente de una base de datos ocho imágenes que ejemplifican conceptos simples u objetos, cada uno de estas imágenes se escala e inserta en una de las ocho particiones aleatorias realizadas con anterioridad en el área de la imagen final. Una vez que se han acoplado todas las imágenes formando una nueva composición, esta será distorsionada aplicando distintos difuminados de colores por regiones seleccionadas de forma aleatoria. La distorsión consigue difuminar los límites entre las imágenes ya que los pixeles adyacentes, aún perteneciendo a imágenes distintas, comparten gama de colores lo que hace más complejo identificar los límites entre las imágenes para una máquina pero no para las personas.

El segundo reto que compone el sistema es un ejemplo típico de CAPTCHA visual, la interfaz consta de una imagen y una lista de palabras de las cuales tan sólo una describirá el objeto o concepto que se muestra. A diferencia de otros CAPTCHAs, la imagen se encuentra distorsionada mediante una combinación de difuminados, particionamientos de la imagen, quantización de los colores, inclusiones de líneas, puntos o mayas de ruido, desdibujamiento de contornos y modificaciones en las gamas de colores.

Figura 16. Ejemplo segunda fase IMAGINATION CAPTCHA

Una de las principales razones por las que los CAPTCHAS basados en imágenes son vulnerables frente a ataques, es que en casi ninguna de las técnicas existentes las imágenes son distorsionadas para evitar que una máquina pueda reconocerlas. La Universidad de Pennsylvania (desarrolladora del sistema IMAGINATION) presenta como solución realizar distorsiones aleatorias sobre las imágenes antes de mostrarlas.

Sin embargo, en la actualidad existen técnicas de reconocimiento de imágenes robustas frente algunas distorsiones, por lo tanto, en este desarrollo se ha diseñado un sistema de distorsiones de imágenes.

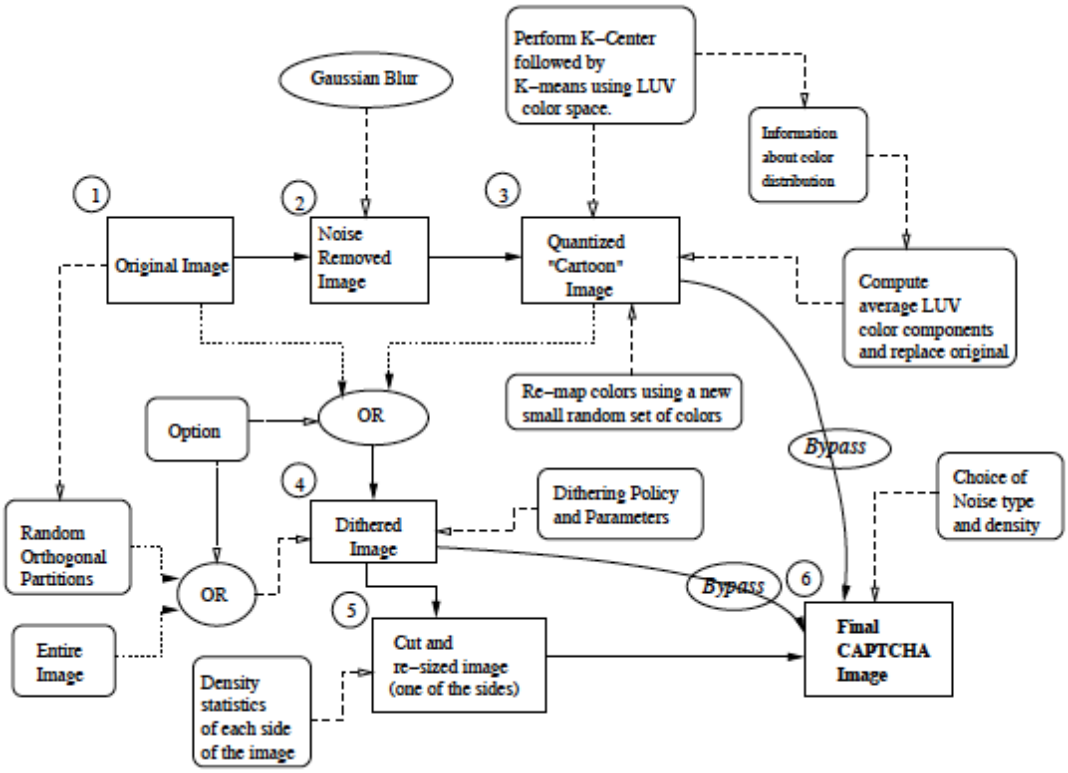

Figura 17. Esquema del sistema de distorsión de IMAGINATION

Simplified 3D to 2D (M. E. Hoque y L. A. Mitchel, 2006)

El esquema de generación de CAPTCHAS de imágenes de 3D a 2D requiere un algoritmo de transformación algo complejo y una amplia base de datos de imágenes, con estos recursos es sencillo encontrar un algoritmo que genere retos sin mucho coste computacional. El test de Turing implementado para diferenciar a las personas de la máquinas se basa en la capacidad que los humanos tenemos en transformar una imagen de 3-D a 2-D, por ejemplo, relacionar un cubo con un cuadrado o conocer que la base de un cilindro es un circulo y su lado un rectángulo.

2.3.3 Aplicación de CAPTCHAs

Un CAPTCHA es un programa que puede generar y evaluar un reto que:

1. Los programas actuales no pueden superar
2. La mayoría de los humanos la pueden pasar

Como sistema, los CAPTCHAs pueden ser usados para diferenciar personas y máquinas y tienen múltiples aplicaciones en el ámbito de la seguridad. Algunas de las aplicaciones más comunes se exponen en esta sección. [3]

Encuestas por internet

En noviembre de 1999, Slashdot realizó una encuesta online para conocer la opinión de los usuarios sobre cuál era la mejor universidad americana de informática, ¡una pregunta peligrosa para realizarla por internet! Como en casi todas las encuestas vía web, la dirección IP de donde proceden los votos se almacenan para prevenir que un usuario pueda votar más de una vez. Sin embargo, los estudiantes de la Universidad de Carnegie Mellon encontraron una forma para amañar las votaciones utilizando programas que realizaban miles de votaciones a favor de su universidad. La puntuación de la CMU empezó a subir muy rápido. Al día siguiente, estudiantes de MIT implementaron su propio programa para conseguir votos, y la encuesta se convirtió en una competición entre los *bots* de ambas universidades. La MIT consiguió 21,156 votos y Carnegie Mellon con 21.032, el resto de universidades no supero los 1000 votos. A partir de esta experiencia apareció la necesidad de crear un mecanismo para evitar que las encuestas por internet fueran amañadas por mediante el uso bots y asegurará que los votos provenían de personas y no de máquinas.

Servicios gratuitos de correo electrónico

Hoy en día muchas empresas proporcionan servicios de correo electrónico gratuito (Yahoo!, Microsoft, Google, etc.), y la mayoría de ellas son vulnerables a ataques de bots que pueden solicitan miles de cuentas de correo en un minuto. Este problema es fácilmente solucionable si se pide que los usuarios se identifiquen como humanos antes de obtener su cuenta. En el año 200, Yahoo! fue una de las primeras empresas que implemento esta medida induyendo en el formulario de petición de cuenta un reto CAPTCHA.

Motores de búsqueda

Los bots de indexación de los motores de búsqueda recorren internet para poder localizar páginas web y añadirlas a sus índices. Sin embargo, existen alguno sitios web que no desean ser incluidos en los motores de búsqueda, estás páginas pueden señalar su preferencia mediante una etiqueta HTML, pero esto no les garantiza que no vayan a ser localizadas y que su código HTML no vaya a ser recorrido por un bot. Para realmente garantizar que la privacidad de esta página no vaya a ser violada por estos bots es necesario recurrir a un CAPTCHA.

E-mails en cadena y spam

Los retos CAPTCHAS también son utilizados como solución para evitar el envió de cadenas de e-mail o spam. De esta forma, sólo son aceptados aquellos e-mails cuyo emisor haya probado que es una persona. Algunas empresas como Spamarrest han explotado esta idea.

Prevenir ataques de diccionario

Una aplicación algo más novedosa es el uso de CAPTCHAs para prevenir los ataques de diccionario de sistemas de autenticación mediante contraseña. La idea es muy sencilla y se basa en evitar que una máquina pueda probar ilimitadamente todas las password que desee. Esta técnica la implementan sistemas tan populares como Twitter que permite realizar tres intentos fallidos al introducir la contraseña, el resto de oportunidades debe de ser validadas cada una de ellas por un CAPTCHA.

2.4 Transformaciones en imágenes

A lo largo de esta sección haremos un resumen del estudio realizado por la Universidad Estatal de Pennsylvania [19] donde realizan una comparativa entre la percepción que los humanos y que las máquinas tienen de imágenes sobre las que se han realizado modificaciones de sus características visuales. El estudio se encuentra enmarcado dentro de los trabajos realizados para el diseño del CAPTCHA IMAGINATION.

Dada la naturaleza del problema es interesante conocer el estado del arte de los algoritmos de procesamiento de imágenes. El estudio se ajusta a la perfección al proyecto, gracias a los resultado obtenidos se puede saber cuáles son las distorsiones apropiadas para aplicar a una imagen para conseguir que no puedas ser reconocidas por las tecnologías actuales y su contenido siga siendo lo suficientemente claro para una persona.

El estado del arte de los algoritmos de reconocimiento de imágenes es todavía bastante primitivo, aún así, algunas técnicas de comparación de imágenes que consiguen avances en este campo. En la actualidad las técnicas de reconocimiento de imágenes son capaces de medir la diferencia entre una imagen natural y una distorsionada, comparar los resultados obtenidos llegando a reconocer si ambas son la misma imagen. Para la realización del estudio se usaron una metodología muy simple de medición de similitudes entre imágenes y dos algoritmos muy conocidos de reconocimiento, ambos utilizan representaciones de las imágenes a bajo nivel y calculan la distancia entre ambas, cada uno de los algoritmos se basa en aspectos diferentes. La primera técnica es la más simple y trata de medir la similitud entre las imágenes calculando la media de los valores de los pixles a lo largo de cada imagen tratando de reconocer un patrón para determinar que ambas son coincidentes, esta técnica es denominada PWD (Pixe-Wise Difference). En segundo lugar se aplico el sistema de reconocimiento de imágenes Earth Mover's Distance (EMD) que basa su medición en la comparación del color global de ambas imágenes. Por último, se utiliza la comparación por IRM que segmenta la imagen y realiza mediciones considerando el color, la textura y las formas que aparecen.

Por otro lado, el estudio de reconocimiento de imágenes por personas se realizo tomando 250 individuos que tuvieron que clasificar imágenes distorsionadas seleccionando la etiqueta adecuada de entre 15 posibilidades. El estudio consiste medir el reconocimiento de imágenes distorsionadas por parte de persona y máquinas de una base de datos de 1050 imágenes clasificadas en 35 categorías.

Las distorsiones elegidas se aplicaron de forma aislada sobre las imágenes del estudio para conocer cómo afectan cada una de ellas y que valores de la distorsión son los adecuados para conseguir que la

imagen sea reconocible para las personas pero no para las máquinas. A continuación se detallan los resultados conseguidos en el estudio.

2.4.1 Brillo

El brillo es una de las propiedades fundamentales que afecta al estado global de la imagen. Aumentar o disminuir la luz ambiental o la luminosidad de una imagen puede llegar a hacer que esta no sea reconocible, por esa razón, es de gran importancia tratar de ajustar sus parámetros correctamente. El componente RGB de cada pixel tiene asignado un valor dentro de la escala de brillo, el valor que toma el brillo asignado a la imagen total se determina realizando la media de los valores de cada uno de los pixeles.

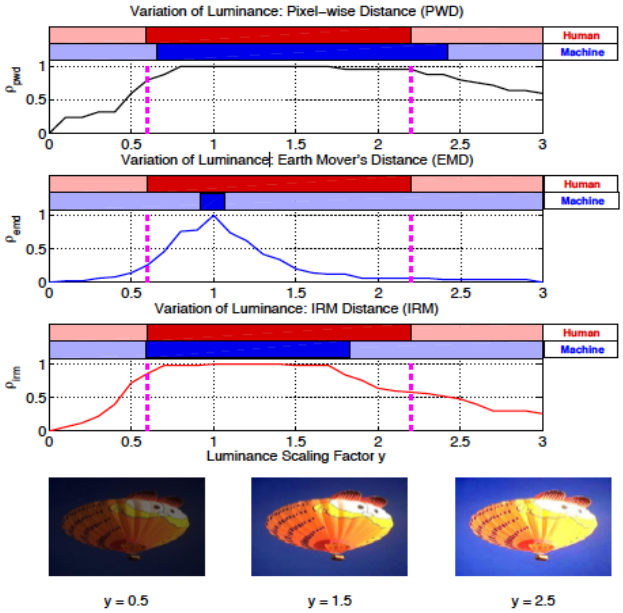

Figura 18 Resultado en la distorsión de brillo

El gráfico de la **Figura** 18 muestra los resultados obtenidos del estudio de la aplicación de distorsiones en el brillo de una imagen. La barra roja muestra que para valores de brillo 0.6 y 2.2 las personas son capaces de reconocer sin gran dificultad la imagen que se les presenta, sin embargo, dependiendo de la técnica de reconocimiento de imágenes empleada los resultados obtenidos por las máquinas son mucho más variables donde los peores resultados los obtiene el algoritmo de EMD.

2.4.2 Cuantización del color

La distorsión mediante cuantización de color limita la escala de colores aplicada en la imagen, el lugar de permitir todos los colores, este tratamiento elige un rango dentro de la escala y realiza una función de transformación sobre todos los pixeles trasladando los valores de los componentes RGB a sus respectivos valores dentro del rango. Esta distorsión normaliza los colores de la imagen haciendo que la

diferencia entre los pixeles sea mucho menor lo que perjudica en gran medida a las técnicas de reconocimiento de imágenes por comparación.

Los parámetros de distorsión para la cuantización se definen mediante niveles, donde, si nivel toma el valor 1 implica que todos los pixeles toman el mismo valor mientras y si el nivel es 15 cada valor RGB de un pixel toma el valor más cercano al original dentro de los quince posibles.

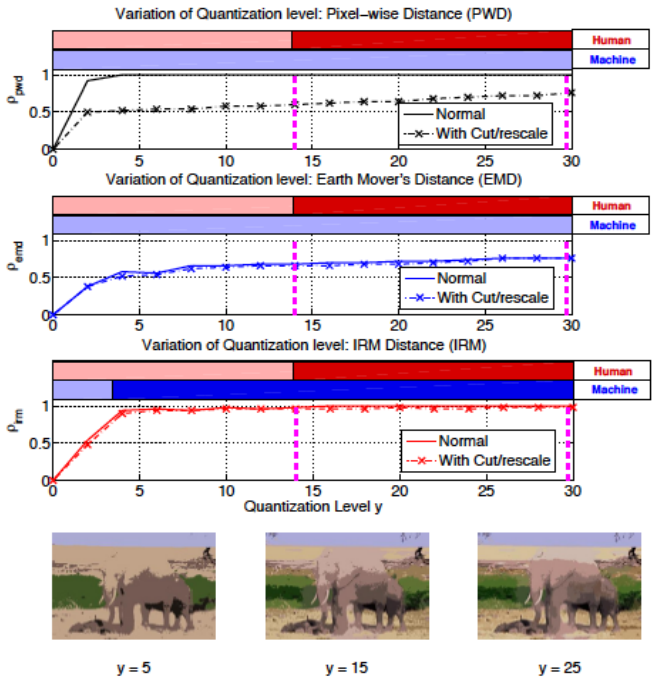

Figura 19. Resultados en la cuantización del color

Los resultados expuestos en las gráficas de la Figura 19 muestra como para las personas a partir de 14 niveles de cuantización todas la imágenes son fácilmente reconocibles a contraposición de la baja probabilidad de acierto que los algoritmos de reconocimiento de imágenes como PWD y EMD, sin embargo, mediante la técnica de IRM el porcentaje es muy alto incluso para valores de cuantización realmente bajos. En el estudio, paralelamente a los resultados recogidos, se han realizado test en las que las imágenes además de ser distorsionadas mediante cuatización han sido cortadas y re-escaladas, los resultados se muestran en la gráficas mediante cruces y tan sólo la técnica de PWD muestra peores resultados debido a esta transformación.

2.4.3 Difuminamiento

La función de difuminamiento es la aproximación digital de la técnica aplicada a las impresiones físicas que transforma algunos colores para crear el efecto óptico de profundidad. Esta técnica, aplicada de forma suave, es muy utilizada en cualquier composición pero una aplicación más agresiva conlleva una distorsión severa de la imagen. Existen múltiples algoritmos y técnicas de aplicación de difuminados, las más comunes se generan mediante formas aleatorias, como cuadros o círculos. En el

estudio realizado por la Universidad de Pennsylvania tan sólo se valora un tipo de difuminación, la técnica particiona aleatoriamente la imagen en bloques ortogonales, como se puede observa en la Figura 20 las imágenes parece que sobre las imágenes se hubieran dibujado rectángulos translucidos, y sobre cada una de estas particiones aplica una gama de colores aleatorias para generar la distorsión. La difuminación, es decir, la gama de colores a aplicar para cada partición, se mide mediante niveles, donde un valor alto indica menor distorsión y un valor pequeño una distorsión mayor.

Los resultados obtenidos muestran como esta transformación es una de las más agresivas hasta ahora, dado que, a niveles medios de distorsión las imágenes obtenidas no son reconocidas por las personas. Aún así, el algoritmo de IRM Distance sigue siendo capaz de reconocer con cierta facilidad la imagen con mayor precisión que las propias personas. La pruebas paralelas en las que además de difuminamiento de la imagen se aplican cortes y re-escalado a estas muestran los mismos resultados que el estudio anterior ya que tan sólo mejoran lo datos si se aplicara una técnica simple de comparación de imágenes.

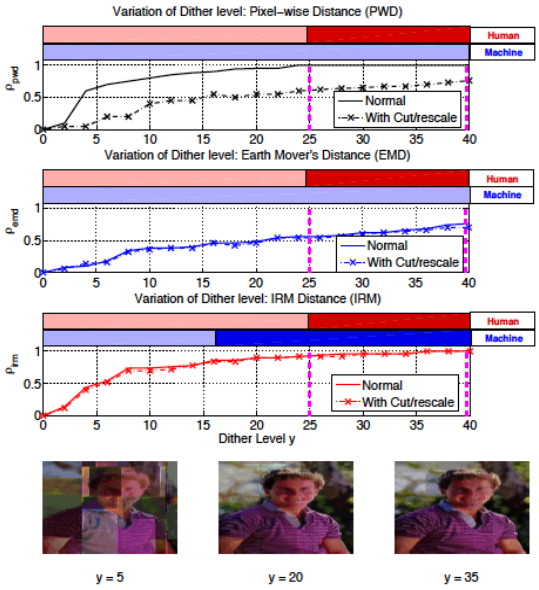

Figura 20. Resultados para el difuminamiento

2.4.4 Ruido por adición de líneas

La inclusión de pixeles de ruido como distorsión de imágenes es una técnica fácilmente reconocida y solventada por muchos de los algoritmos de reconociendo de imágenes, en su lugar, se propone incluir a la imagen un número de líneas ya sean rectar o curvas o superponer una maya a la totalidad de la imagen. Esta trasformación supone una distorsión muy agresiva que modifica en gran medida la composición final pero que no afecta a la percepción de la personas debido a la capacidad que estas tienen de poder completar huecos u omitir ciertas partes de una composición. Las líneas incluidas se seleccionan de forma aleatoria, así como el color de estas, aunque aparentemente el RGB toma valores cero (negro), para reducir los aciertos de los métodos de detecciones y eliminación de ruido, el color se elige mediante un valor aleatorio. La distorsión generada se mide mediante la densidad de líneas, siendo y el espacio que existe entre las líneas generadas.

La Figura 21 muestra la relación entre la inclusión de líneas y la dificultad que esta distorsión supone tanto para las personas como para las máquinas. A diferencia del resto de estudios, el método de detección de imágenes que mejores resultados obtiene es el algoritmo de PWD que alcanza un gran porcentaje de aciertos.

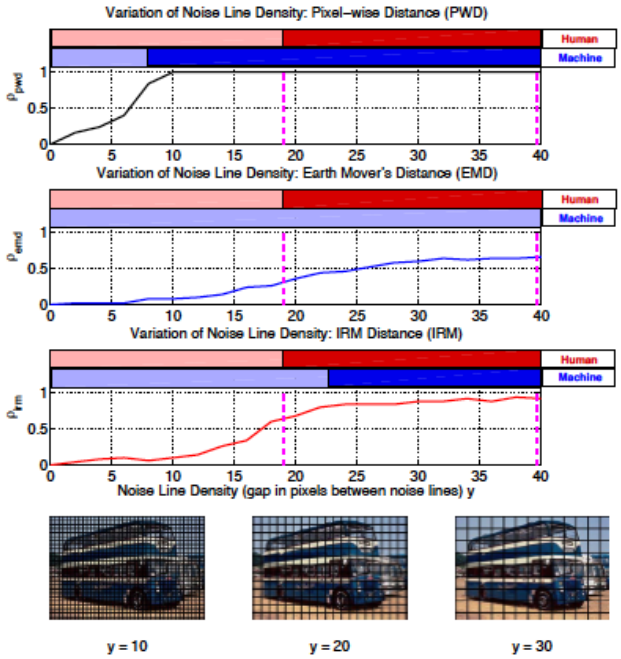

Figura 21. Resultados en la inclusión de ruido

2.4.5 Conclusiones

Feature	Affected by	Not Affected by
Local Color	Quantization, Dithering, Luminance, Noise	Cut/rescale
Color Histogram	Luminance, Noise, Cut/rescale	Quantization, Dithering
Texture	Quantization, Dithering, Noise	Luminance, Cut/rescale
Edges	Noise, Dithering	Quantization, Luminance, Cut/rescale
Segmentation & Shape	Dithering, Noise, Quantization	Luminance, Cut/rescale
Interest Points	Noise, Dithering, Quantization	Luminance, Cut/rescale

Figura 22. Resumen estudio tranformaciones en imágenes

Los resultados obtenidos recogen datos sobre el comportamiento de las distorsiones que potencialmente podrán ser aplicadas para la modificación de imágenes. Aunque ninguna de las distorsiones por separado y de forma atómica satisface los requisitos deseados, este estudio aporta una gran información sobre los parámetros de aplicación y los aspectos que se ven afectados por cada una de ellas.

2.5 Tarjetas inteligentes

2.5.1 Introducción

Una tarjeta inteligente es una tarjeta microprocesadora de las dimensiones de una tarjeta de crédito (o más pequeña, como por ejemplo, tarjetas SIM o GSM) con varias propiedades especiales (ej. un procesador criptográfico seguro, sistema de archivos seguro, características legibles por humanos) y es capaz de proveer servicios de seguridad (ej. confidencialidad de la información en la memoria). "Las tarjetas inteligentes nacieron a principios de los 80 y su filosofía es muy sencilla, tratan de almacenar información con una cierta autonomía y seguridad. Aunque la cantidad de información que pueden almacenar es relativamente pequeña, sus capacidades son lo suficientemente importantes como para haber producido la expansión de este tipo de tarjetas en el mercado".

Figura 23. Tarjeta Inteligente

Existen dos tipos de tarjetas, las de memoria y las de microprocesadores.

- **Las tarjetas de memoria:** son las más comunes de hallar en aplicaciones comerciales como tarjetas de prepago. Este tipo de tarjeta funciona como un simple almacenador de información que el usuario modifica cuando realiza una transacción con ella. Es así por ejemplo, como en el caso de la telefonía en el que la tarjeta viene de fábrica con el contenido de minutos que el usuario puede ocupar. Al hacer una llamada con la tarjeta, la máquina en cada minuto va descontando uno de los minutos que trae la tarjeta, de esa manera se evita que el usuario se sobregire. Todos los tipos de tarjetas inteligentes deben incorporar algún tipo de memoria. Hasta el momento los que emplean son:
    - ROM memoria de lectura solamente.

- PROM memoria de lectura solamente y programable.
- EPROM una ROM programable y borrable.
- EEPROM una PROM borrable eléctricamente.
- RAM memoria de acceso aleatorio.

- **Tarjetas de Procesamiento:** En esta clase de tarjetas inteligentes caen todas aquellas tarjetas que en su interior traen un microchip que puede procesar la información que almacenan las tarjetas. Para el caso de autentificación un usuario podría tener en la memoria de la tarjeta inteligente el fondo de su retina. Cuando se quiera autentificar solo tendría que mostrar su retina y contrastarla con la que viene en la tarjeta. Algunas de sus características son:
  - Las tarjetas inteligentes que incorporan el microchip pueden adherir, borrar y de alguna manera manipular información en su memoria. Pueden ser vistas como un computador en miniatura con un puerto de entrada/salida, sistema operativo y disco duro.
  - El microchip trae un microprocesador que está disponible en arquitecturas de 8, 16 y 32 bits.
  - Su capacidad de almacenamiento de datos varía entre 300 a 32000 bytes con expectativa de incrementar esto último con los avances tecnológicos.

Las tarjeta inteligentes pueden ser de contacto o sin contacto. Las que no requieren contacto solamente necesitan aproximarse al lector y la comunicación se hace a través de una enviándose ondas como las de radio. Las otras poseen contactos eléctricos situados en el exterior de la tarjeta conectan con un lector de tarjetas cuando se inserta la tarjeta. Se espera que en un futuro muy cercano todas las transacciones se realicen mediante este tipo de tarjetas reemplazando finalmente a las tarjetas de banco, de afiliación y de transporte entre otras. “Las Smart Cards han existido en varias formas desde 1974. Desde ese entonces, gracias a la motivación de compañías como Gemplus y Schlumberger han recibido gran atención en el mercado de los dispositivos de control. Según la consultora Frost & Sullivan más de 600 millones de SmartCards fueron emitidas en 1996 y se espera un consumo de 21 billones para el año 2010”.

2.5.2 Aplicaciones de las tarjetas inteligentes

Las tarjetas inteligentes tienen diferentes usos y aplicabilidades en las diferentes esferas de negocios. Entre las diferentes aplicaciones en la industria sobresalen las de identificación de personas, control de acceso, tarjeta para clientes, entre otras.

Industria	Aplicación
Contabilidad	Tarjetas comerciales, tarjetas de identificación de clientes, tarjetas de promociones, tarjetas de calendario itinerario.
Aeropuertos	Tarjetas para el acceso a empleados, credenciales con identificación para seguridad.

Miembros de asociaciones	Tarjetas de identificación, tarjetas para descuento en puntos de venta, tarjetas calendario.
Distribuidores de automóviles	Tarjetas de identificación del vehículo, tarjetas de fidelidad de distribuidores, tarjetas de descuento, tarjetas de garantía.
Bares, clubes nocturnos	Tarjetas VIP, tarjetas para entrada, tarjetas para miembros.
Lavadero de automóviles	Tarjetas de frecuencia de uso, tarjetas de prepago de lavados.
Clubes	Tarjetas de miembros.
Ordenadores	Tarjetas de garantía, soporte a clientes, número de accesos a Internet, descuentos.
Tintorerías/lavanderías	Tarjetas de descuento, tarjeta de frecuencia de clientes.
Hoteles	Tarjetas de descuentos, tarjetas de frecuencia, tarjetas llave, tarjetas de identificación de equipajes.
Inversión	Tarjetas de clientes, tarjetas calendario.
Biblioteca	Tarjetas de identificación, códigos de barra.
Bienes raíces	Tarjetas de negocios, tarjetas telefónicas, tarjetas calendario.
Servicio de alquiler	Identificación, entrada preferencial.
Comercio	Tarjetas de clientes, tarjetas de control de pagos, tarjetas de control, tarjetas de fidelidad.
Seguridad	Control de acceso, name badges.
Centros comerciales	Clientes, tarjetas de descuento, programas *legales*.

Tabla 1. Aplicación tarjetas inteligentes

2.5.3 Tecnología JavaCard

Java Reconocido lenguaje de la empresa Sun Microsystem paso a ser una tecnología utilizada para brindar soluciones en todos los campos de la informática, su versatilidad, portabilidad y seguridad la ha convertido en la tecnología ideal para sus diferentes tipos de aplicaciones, desde centros de datos, de consolas de juegos a súper quipos científicos, de teléfonos móviles a Internet, Java está en todos lados. "Hasta la fecha, la plataforma Java ha atraído a más de 6,5 millones de desarrolladores de software. Se utiliza en los principales sectores de la industria de todo el mundo y está presente en un gran número de dispositivos, equipos y redes".

Una de las vertientes de la tecnología Java es Javacard., la cual consiste en un conjunto de especificaciones basadas en la plataforma java que pueden ejecutarse dentro de una Smart Card. La tecnología de JavaCard permite a las smarts card y a otros dispositivos con memoria muy limitada ejecutar programas pequeños, llamados applets, que emplean la tecnología de Java. Con esta se provee a los fabricantes de tarjetas inteligentes de una plataforma segura e interoperable que pueda almacenar y actualizar usos múltiples en un solo dispositivo. La tecnología de JavaCard es compatible con estándares existentes de tarjetas inteligentes.

Esta tecnología permite a los desarrolladores diseñar, construir, evaluar e implementar aplicaciones y servicios de forma rápida y segura., reduciendo costos, incrementar la productividad y generando un valor agregado al cliente.

Java Card incluye la especificación de la plataforma Java Card y el kit de desarrollo de Java Card. La primera define las características, servicios, y comportamiento que deben soportar las implementaciones e incluye la Java Card Virtual Machine (VM), el entorno de desarrollo Java Card y las librerías de la plataforma. La segunda está conformada por un conjunto de herramientas para el desarrollo e implementación de aplicaciones JavaCard, emuladores, simuladores y la documentación de la plataforma.

A la hora de realizar una aplicación real, con tarjetas smart card es importante escoger una tarjeta que incluya todas las funcionalidades de java, como el API, la maquina virtual y algunos recursos de software. En el mercado actualmente es posible ubicar todas las piezas para armar una aplicación real. "Todo el mundo puede montar su propio kit de desarrollo, la compra de cada uno de los elementos (lector de tarjetas inteligentes, tarjetas inteligentes, y SDK Java applet Tarjeta cargador) de diferentes fabricantes o, en algunos casos, obtener algunos de estos elementos de forma gratuita."

Capítulo 3

ESPECIFICACIÓN DE REQUISITOS USUARIO

3.1 Especificaciones generales

3.1.1 Capacidades generales

Se pretende el desarrollo de una solución que permita una comunicación segura entre una persona y una tarjeta inteligente. A lo largo de este proyecto, se diseñará e implementará una solución que asegure que las funcionalidades de una tarjeta inteligente sean utilizadas de una forma legítima aunque el terminal que actúa como intermediario no sea confiable. Esta solución permitirá que una persona pueda percatarse cuando el terminal donde ha insertado su tarjeta inteligente está haciendo un uso fraudulento de esta.

Se desarrollará un sistema que permita realizar firmas digitales de un mensaje con una tarjeta inteligente en entornos no confiables sin que la integridad de la petición de firma se vea afectada por cualquier programa malicioso que se encuentre activo en dicho entorno. El objetivo principal del proyecto se centrará en encontrar una solución que permita que sea el propio usuario el verificador de

la integridad de la petición. El dispositivo criptográfico no llevará a cabo ninguna operación sin antes verificarla y obtener una confirmación explicita por parte del usuario de la petición.

Dado que las comunicaciones entre el usuario y su tarjeta inteligente se realizan a través de un medio hostil, ambas partes deben poder comprobar la autoría de la información que reciben. La tarjeta inteligente deberá construir el mensaje de verificación de la petición de firma de tal manera que cualquier persona, sin necesidad del uso de mecanismos criptográficos extras, pueda comprender su contenido y autenticarla como el emisor. De esta forma, si el usuario observa que el mensaje de verificación no se ajusta a los parámetros de su petición o no puede autenticar como emisor a la tarjeta inteligente tendrá conocimiento de que existe algún proceso malicioso que intenta realizar un ataque sobre la petición de firma. Por otro lado, también aparece la necesidad de que la solución asegure que la confirmación del usuario únicamente haya podido ser enviada por una persona, es decir, que en ningún caso una máquina pueda dar una confirmación válida y suplantar de esta forma al usuario.

3.1.2 Restricciones generales

La solución desarrollada debe basarse en la utilización de una tarjeta inteligente como dispositivo criptográfico y almacenamiento de información de autenticación de usuario que debe cumplir los estándares PKCS.

La tarjeta inteligente estará asociada exclusivamente a una única persona. Además, este dispositivo criptográficos debe estar previamente formateado, es decir, contener las claves privada y pública del usuario, su certificado digital y los mecanismo necesarios para llevar a cabo las operaciones tanto propias de su funcionalidad como de la solución a desarrollar en este proyecto.

El sistema debe permitir que el usuario introduzca sus mensajes, realice las peticiones y verifique estas mediante la entrada estándar del sistema. En ningún momento el usuario necesitará de ningún dispositivo extra para poder comprender el contenido del mensaje de verificación o para generar la confirmación de dicho mensaje.

3.1.3 Características de los usuarios

Se identifica un único perfil de usuario para el sistema a desarrollar. El usuario es una personal con conocimientos básicos de seguridad, sabe que es una firma digital y como utilizarla, dispone de un dispositivo criptográfico donde está almacenado su certificado digital, una tarjeta inteligente, pero no siempre dispone de un terminal confiable para realizar la firma digital de un mensaje. Este usuario deberá estar familiarizado con el uso de tarjetas inteligentes y debe conocer su PIN personal para realizar el login.

Sin embargo, para este desarrollo se presupone que las tarjetas inteligentes se encuentran previamente formateadas, es decir, contienen la información necesaria para llevar a cabo las operaciones criptográficas pertinentes y se han instalado los mecanismos que permiten realizar la firma digital según las premisas declaradas en los requisitos. Esta fase previa de instalación será llevada a cabo por un administrador, pero en ningún momento se incluirá estas capacidades dentro del desarrollo del proyecto.

3.1.4 Entorno operacional

Para que una persona pueda interactuar con su tarjeta inteligente es necesario que lo realice a través de un terminal que disponga de un lector de tarjetas. El terminal recibe las peticiones del usuario y las transforma en instrucciones comprensibles para la tarjeta inteligente y a su vez debe interpretar para el usuario las respuestas en lenguaje máquina que devuelve la tarjeta inteligente.

Figura 24. Entorno operacional

En el planteamiento del problema, descrito en la sección **¡Error! No se encuentra el origen de la referencia.**, se plantea un escenario básico en el que se hace uso de un terminal público adherido a una red. Este será el entorno operacional de sistema a diseñar.

El terminal estará provisto de dispositivo de entrada y salida de datos estándar, es decir, pantalla y teclado; estará equipado de un lector de tarjetas y, además, tendrá acceso a Internet.

En general la solución tiene que ser aplicable en terminales donde el que el usuario no tiene ningún tipo de control sobre los programas que se encuentran en ejecución y donde no tiene conocimiento de los intereses por los que actúa el terminal.

3.2 Requisitos de usuario

Un requisito de usuario es una especificación de lo que debe hacer el sistema y la forma de hacerlo. Describe el *qué* hacer, pero no el *cómo*. El gran reto en la obtención de los requisitos es saber entender, capturar y expresar con claridad las necesidades que tiene el cliente y que han de ser resueltas mediante la futura aplicación software. Los requisitos pueden dividirse en dos grupos principales, según sea su naturaleza:

- **Requisito de capacidad:** funciones y operaciones requeridas por los usuarios para resolver un problema o alcanzar un objetivo. Describe una operación, o secuencia de operaciones, que el software debe ser capaz de realizar.

- **Requisito de restricción:** restricciones impuestas por los usuarios sobre la manera en que el problema es resuelto o el objetivo es alcanzado; restringe la manera en que el software es construido o funciona, sin alterar o describir las capacidades del software.

Cada requisito se ha definido en una tabla que contiene los siguientes campos:

3. **Identificador:** el identificador de cada requisito debe ser tal que lo referencia de forma unívoca frente a los demás. Su formato será del tipo:
   - *RUC-XXX:* Se usará para identificar los requisitos de usuario de capacidad (RUC).
   - *RUR-XXX:* Se usará para identificar los requisitos de usuario de restricción (RUR).

   Donde XXX representa un número entre 001 y 999, siendo el 001 el primer número asignado a un Requisito y que crecerá de manera secuencial.
4. **Nombre:** Nombre específico del requisito.
5. **Descripción:** representa una descripción textual de la naturaleza y objetivo del requisito. La descripción ha de ser breve, concisa y sin ambigüedad.
6. **Prioridad:** Medida de la prioridad del requisito con el fin de establecer una estrategia de desarrollo. Determina la importancia de cumplir con el requisito antes de abordar otros requisitos o actividades productivas. Su valor puede ser:
   - *Alta*: máxima prioridad en abordar el cumplimiento requisito.
   - *Media*: el requisito es importante, pero no es el más prioritario.
   - *Baja*: el requisito no es prioritario.
7. **Fuente**: especifica el origen del requisito (una determinada metodología, una especificación directa del cliente, etc.).
8. **Necesidad:** la necesidad marca cómo de importante y útil es el requisito dentro del sistema y su influencia en la plena satisfacción del cliente. La necesidad de un requisito puede ser:
   - *Esencial*: el requisito ha de ser cumplido obligatoriamente.
   - *Deseable*: el requisito no es obligatorio, pero sí aportaría mayor calidad y satisfacción al cliente.
   - *Opcional*: el requisito puede no ser cumplido sin influir negativamente en la satisfacción final.
9. **Estabilidad:** indica cómo de probable es que el requisito pueda ser modificado en el futuro, o incluso eliminado. Los valores de este atributo pueden ser:
   - *Alta*: el requisito no va a ser nunca modificado.
   - *Media*: el requisito puede ser modificado en algún momento del desarrollo.
   - *Baja*: el requisito será modificado con seguridad durante el desarrollo.
10. **Dependencias:** Identificadores de otros requisitos con los que éste guarda algún tipo de relación.

3.2.1 Requisitos de capacidad

Identificador	RUC-001	**Dependencias**	No aplica
Nombre	Autenticación del usuario		
Descripción	Autenticar unívocamente al propietario/usuario de la tarjeta inteligente en el proceso de login.		

Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Cliente
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 2. RUC-001

Identificador	RUC-002	**Dependencias**	No aplica
Nombre	Firmar mensaje		
Descripción	Realizar la firma digital sobre un mensaje introducido por el usuario.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Cliente
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 3. RUC-002

Identificador	RUC-003	**Dependencias**	No aplica
Nombre	Verificación del mensaje		
Descripción	Verificar unívocamente el mensaje antes de firmarlo.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Cliente
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 4. RUC-003

Identificador	RUC-004	**Dependencias**	No aplica
Nombre	Finalizar operación		
Descripción	Finalizar operación a petición del usuario		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Cliente
Necesidad	☑ Esencial ☐ Deseable	**Estabilidad**	☑ Alta ☐ Media

	☐ Opcional		☐ Baja

Tabla 5. RUC-004

Identificador	RUC-005	**Dependencias**	No aplica
Nombre	Fin automático de la operación		
Descripción	Finalizar automáticamente la operación al desconectar la tarjeta.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Cliente
Necesidad	☐ Esencial ☑Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 6. RUC-005

3.2.2 Requisitos de restricción

Identificador	RUR-001	**Dependencias**	RUC-002
Nombre	Tarjeta Inteligente		
Descripción	La firma de los mensajes se hará con una tarjeta inteligente.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Cliente
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 7. RUR-001

Identificador	RUR-002	**Dependencias**	No aplica
Nombre	Restricción autenticación		
Descripción	El usuario deberá autenticarse para hacer la petición de firma. La autenticación consistirá en introducir el número PIN de la tarjeta inteligente.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Cliente

Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 8. RUR-002

Identificador	RUR-003	**Dependencias**	No aplica
Nombre	Código PIN		
Descripción	El PIN de la tarjeta inteligente estará compuesto por 5 dígitos.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Cliente
Necesidad	☐ Esencial ☑ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 9. RUR-003

Identificador	RUR-004	**Dependencias**	No aplica
Nombre	Bloqueo tarjeta		
Descripción	La tarjeta inteligente se bloqueará tras la inserción del número PIN erróneo durante 3 veces consecutivas.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Cliente
Necesidad	☐ Esencial ☑ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 10. RUR-004

Identificador	RUR-005	**Dependencias**	No aplica
Nombre	Formato tarjeta inteligente		
Descripción	La tarjeta inteligente habrá sido previamente formateada con la información y mecanismos necesarios para realizar la firma digital de un mensaje.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Cliente

Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 11. RUR-005

Identificador	RUR-006	**Dependencias**	No aplica
Nombre	Restricción usuario tarjeta inteligente		
Descripción	Una tarjeta inteligente sólo será válida para una única persona.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Cliente
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 12. RUR-006

Identificador	RUR-007	**Dependencias**	RUC-003
Nombre	Restricción mensaje verificación		
Descripción	El usuario debe autenticar como emisor del mensaje de verificación a la tarjeta inteligente sin el uso de ningún dispositivo extra.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Cliente
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 13. RUR-007

Identificador	RUR-008	**Dependencias**	RUC-003
Nombre	Restricción confirmación de petición		
Descripción	La confirmación sólo podrá ser generada por una persona.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Cliente
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 14. RUR-008

Identificador	RUR-009	**Dependencias**	RUC-002
Nombre	Mensaje		
Descripción	El mensaje será una cadena de caracteres alfanuméricos introducidos por teclado.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Cliente
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 15. RUR-009

Identificador	RUR-010	**Dependencias**	RUC-005, RUC-006
Nombre	Restricción finalización de operación		
Descripción	La operación deberá estar inicializada previamente.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Desarrollador
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 16. RUR-010

Identificador	RUR-011	**Dependencias**	No aplica
Nombre	Lector de tarjeta		
Descripción	El terminal deberá estar provisto de un lector de tarjetas inteligentes		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Cliente
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 17. RUR-011

3.3 Modelo de casos de uso

El modelado de caso de uso es una técnica para la captura de requisitos potenciales de un nuevo sistema o una actualización de software. Cada caso de uso proporciona uno o más escenarios que indican cómo debería interactuar el sistema con el usuario o con otro sistema para conseguir un objetivo específico.

Los elementos básicos que participan en el modelo de casos de uso son:

- **Actor:** es una persona, organización o sistema externo que desempeña el papel en una o más interacciones con el sistema con el de lograr un objetivo, es decir, es el usuario del sistema. También se consideran actores todo aquello que inicia un caso de uso.

- **Caso de uso:** es lo que pasa cuando el actor interactúa con el sistema con el deseo de lograr un objetivo. Se describe normalmente comenzando con un verbo que representa la acción.

- **Asociación:** es la relación entre un actor y un caso de uso, o entre dos casos de uso. Este último caso se da cuando un caso de uso incluye otro, extiende a otro o generaliza a otro.

- **Escenario:** es un camino que puede tomar un caso de uso. Existen escenarios exitosos, en los cuales el objetivo del caso de uso se logra, y los escenarios fallidos, donde el objetivo nos e logra. Un caso de uso puede tener varios escenarios posibles.

De forma más específica, se puede definir un caso de uso como una secuencia de interacciones que se desarrollarán entre un sistema y sus actores en respuesta a un evento que inicia un actor principal sobre el propio sistema. Los diagramas de casos de uso sirven para especificar la comunicación y el comportamiento de un sistema mediante su interacción con los usuarios y/u otros sistemas.

3.3.1 Diagrama UML de casos de uso

Un diagrama de casos de uso muestra la relación entre los actores y los casos de uso del sistema. Representa la funcionalidad que ofrece el sistema en lo que se refiere a su interacción externa.

El Lenguaje Unificado de Modelado (UML) provee de un grupo de elementos gráficos para representar un caso de uso, de manera explícita, suscita y esquemática. El estándar utiliza un rectángulo para representar el sistema a desarrollar. En el interior del sistema deben aparecer los casos de uso, cada uno de ellos consiste en una elipse con una leyenda que indica el nombre del mismo. En el exterior del sistema se sitúan los actores, representados como unos monigotes. Las línea rectas son asociaciones, y esta puede darse entre un actor y un caso de uso o entre un par de casos de uso.

En la

Figura 25 podemos observar la representación del diagrama de casos de uso del sistema que se desarrollará. Es un diagrama muy sencillo que contiene un único caso de uso genérico que deriva en otros dos más específicos. El coste del desarrollo de una solución para del sistema requerido no se basa en la funcionalidad de este sino en la complejidad e innovación necesarios para alcanzar los requisitos.

Figura 25. Modelo Casos de Usos

3.3.2 Descripción formal

Cada caso de uso se ha definido en una tabla que contiene los siguientes campos:

1. **Identificador:** el identificador de cada caso de uso debe ser tal que lo referencia de forma unívoca frente a los demás. Su formato será del tipo:
   - *CU-XXX:* Se usará para identificar los casos de uso (CU).

   Donde XXX representa un número entre 001 y 999, siendo el 001 el primer número asignado a un Requisito y que crecerá de manera secuencial.
2. **Nombre:** nombre específico del caso de uso.
3. **Descripción:** representa una descripción textual de la naturaleza y objetivo del caso de uso. La descripción ha de ser breve, concisa y sin ambigüedad.
4. **Actores:** nombre de los actores primarios y secundarios que interaccionan con el caso de uso.
5. **Precondiciones**: condiciones que debe satisfacer el sistema para poder ejecutar el caso de uso.
6. **Postcondiciones:** condiciones que debe satisfaces el sistema al finalizar la ejecución del caso de uso.
7. **Escenario básico:** secuencia de iteraciones entre el actor y el sistema, representa el flujo de eventos más importantes del caso de uso.
8. **Escenarios alternativos:** representan los diferentes flujos de eventos que podría tomar el caso de uso.

Identificador	CU-001	**Dependencias**	No aplica
Nombre	Firmar mensaje		
Descripción	El terminal deberá estar provisto de un lector de tarjetas inteligentes		
Actores	Usuario		
Precondiciones	El usuario debe tener una tarjeta inteligente correctamente formateado conocer su clave de acceso		
Postcondiciones	Se debe crear un fichero con el mensaje y su firma		
Escenario básico	1. Autenticarse con el PIN de la tarjeta 2. Introducir por teclado el mensaje 3. Mostrar mensaje de verificación con información de la petición de firma 4. Introducir código confirmación 5. Indicar directorio para almacenar el fichero con el mensaje y la firma 6. Obtener mensaje con firma 7. Salir del sistema		
Escenarios alternativos	1-1a. Autenticación es fallida tres veces 7. Salir del sistema		
	3-3a. Información de mensaje en la verificación no es correcta 7. Salir del sistema		
	4-4a. Código confirmación fallido 3 veces 7. Salir del sistema		

Tabla 18. CU-001

3.3.3 Diagrama de actividad

Para facilitar la comprensión de los casos de uso, se han realizado el correspondiente diagrama de actividad.

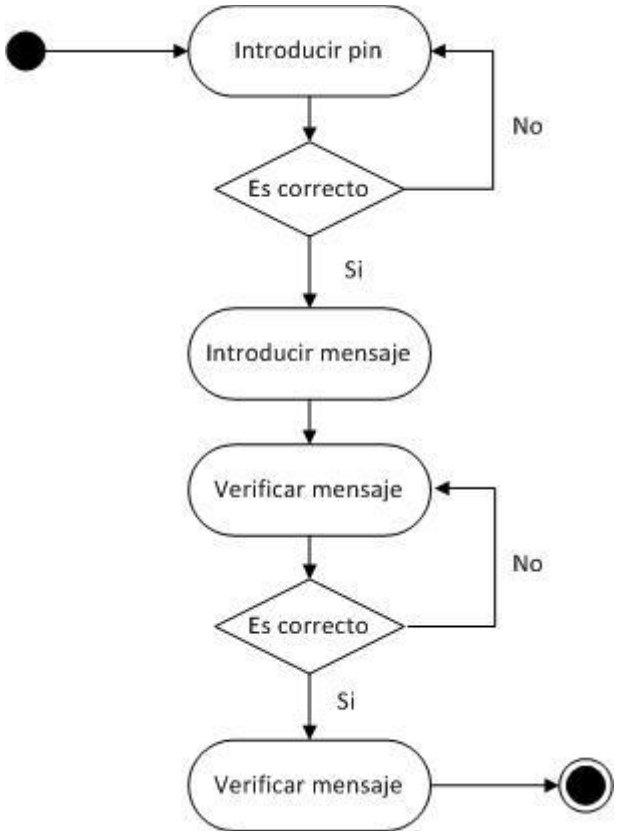

Figura 26. Diagrama de actividad

Capítulo 4

DISEÑO

4.1 Descripción del modelo

En esta sección se procederá a describir la solución que se ha ideado para resolver las necesidades descritas en los requisitos de usuario del capítulo 0, donde, se proponía la construcción de un protocolo de seguridad que permita que cualquier persona pueda conocer, en todo momento, las operaciones que está realizando su tarjeta inteligente aunque el sistema a través del que se comunican no sea totalmente confiable.

4.1.1 Esquema lógico de la solución

La solución ideada para alcanzar los objetivos marcados en este proyecto sigue un esquema reto-respuesta que permitirá realizar una autenticación bidireccional. Se pretende construir un sistema que al recibir una petición de firma genere un mensaje de verificación que enviará al usuario (reto) y este, si los parámetros son correctos, debe enviar una confirmación (respuesta). El protocolo deberá cumplir tres características básicas:

- El mensaje de verificación debe ser integro.
- El usuario debe poder identificar al emisor del mensaje de verificación al sistema.

- El sistema debe poder identificar al emisor de la confirmación al usuario.

Las técnicas implementadas en el diseño del mecanismo de seguridad que se utilizará como mensaje de verificación están basadas en las características de un problema complejo de la Inteligencia Artificial. Este problema se utilizará a modo de función de seguridad. Cuando en la inteligencia artificial se habla de problemas "AI-complete" o "AI-hard" se hace referencia a aquellos problemas que su resolución implica una complejidad computacional equiparable a la realizada por la mente humana. La razón por la que este tipo de problemas resultan interesantes en el ámbito de la seguridad es debido a que un problema muy complejo para la inteligencia artificial puede ser resuelto de forma rápida y sencilla por la mayoría de las personas, pero no por una máquina que utilice las técnicas actuales del estado del arte de la inteligencia artificial. "Los criptógrafos vagos utilizan técnicas de inteligencia artificial." [6].En nuestro caso emplearemos como función criptográfica las técnicas aplicadas al test de Turing invertido por excelencia, el CAPTCHA.

La idea es sencilla. Dada un dispositivo confiable, como puede ser una tarjeta inteligente, que desea proteger el contenido de un mensaje dirigido a un receptor humano al enviárselo a través de un medio hostil, si el mensaje se transforma con las técnicas utilizadas en la construcción de restos de tipo CAPTCHA, se proveerá a este de confidencialidad e **integridad** y lo hará invulnerable a ataques realizados por otras máquinas. En lugar de generar un mensaje aleatorio, como usualmente se hace en la generación de CAPTCHAs, el sistema transformará el mensaje que se desea firmar y se lo enviará al usuario para que el pueda verificarlo. La aplicación de esta técnica nos asegura que una máquina externa al sistema no podrá conocer el contenido del mensaje de verificación, ni realizar modificaciones sobre él, sin embargo, un atacante podría simplemente reemplazar por completo el mensaje.

Dado que los parámetros de la transformación son públicos, una función criptográfica no puede basarse en el secreto de su algoritmo sino en la fuerza de este, no existe ningún aspecto que identifique como emisor del mensaje de verificación al sistema. Un mensaje con las características anteriores puede ser generado por cualquier máquina, no solo por la plataforma confiable. El problema es que no existe una relación ente el mensaje de verificación y la identidad del mismo. Para resolverlo se introduce una clave compartida entre el usuario y el sistema y se acordara previamente al uso del protocolo entre ambas partes. El algoritmo de generación del mensaje de verificación recibirá como entrada un mensaje *m* y una clave secreta *k*, y devolverá un nuevo mensaje *t*. La clave se incluirá en el mecanismo de verificación mediante transformaciones que también se basan en problemas complejos de la IA. Las personas, a diferencia de las máquinas, tienen la capacidad realizar asociaciones y clasificar objetos en categorías. Aprovechando esta característica del ser humano se puede determinar una clave que no sea un objeto estático sino un concepto, por lo que su representación puede variar haciendo más compleja su identificación para un máquina atacante, pero sin aumentar la dificultad de la verificación. Para una persona es sencillo recordar la clave y comprobar si se ha usado en el mensaje. El sistema podrá identificarse mediante este concepto secreto enviando al usuario imágenes que contengan representaciones del concepto. Para elevar la seguridad del mecanismo, el sistema generará una imagen diferente para cada verificación, aunque se siga utilizando el mismo concepto como clave compartida. Por ejemplo, si el usuario y el sistema han acordado que su secreto compartido es automóvil, para cada autenticación se escogerían imágenes diferentes, todas las ilustraciones que aparecen en la Figura 27 podrían ser válidas ya que hacen referencia al concepto utilizado como clave.

Figura 27. Colección de imágenes de automóviles

Creemos conveniente que el secreto se transmita mediante imágenes ya que otras opciones barajadas, como utilizar ficheros de audio donde, en cada ocasión, una voz distinta pronuncien el secreto implica problemas básicos de recursos y medios (es necesario que el usuario disponga de software especifico de reproducción de audio, es poco viable en entornos ruidosos, etc.). Si para incluir el mensaje en el mecanismo de verificación se utilizaban transformaciones de CAPTCHAs de texto, se aplicarán transformaciones empleadas en las técnicas de generación de CAPTHAs de imágenes para integrar la clave en el mecanismo, manteniendo de esta forma las propiedades de confidencialidad e integridad, que por las características de las transformaciones ya tenía el mensaje de verificación, y añadiéndole la capacidad de autenticación.

Una de las técnicas más comunes para la elaboración de CAPTCHAS es la inclusión de ruido para que las máquinas atacantes tenga mayor dificultad en determinar qué información es importante y cual desechable. Siguiendo esta idea en los CAPTCHAS basados en imagen es habitual la inclusión de líneas o máscaras, otra opción es la indusión de más de una imagen. Teniendo en cuenta los recursos de procesamiento de imágenes se ha decidido incluir una imagen, además de la que incorpora la clave, de temática aleatoria para que la vulnerabilidad del mecanismo de verificación sea menor ante los avances de las técnicas de reconocimiento de imágenes.

Una vez recibido el mensaje de verificación, el usuario deberá confirmar si el mensaje es el adecuado. Un primer acercamiento a la solución era hacer que el usuario introdujera su número PIN como confirmación para la acción de firma. Sin embargo, en la secuencia de pasos para hacer uso de las funcionalidades de la tarjeta ya se ha requerido el PIN para acceder al sistema, por lo tanto, podría existir un proceso espía malicioso en el terminar que hubiera recogido este dato y podría mandar confirmaciones a la tarjeta sin necesidad de comprender el mensaje de verificación. Por esta razón se ve necesario integrar la confirmación dentro del mensaje de verificación haciendo uso de las características de los retos CAPTCHA, de esta forma, el sistema podría estar seguro de que tan sólo una persona es capaz de generar una confirmación correcta. La confirmación consistiría en una clave de un único uso, más conocida como *one time password* u OTP. La tarjeta inteligente, al recibir el mensaje que el usuario desea firma, generaría un número aleatorio, mediante sus capacidades criptográficas, que incluiría junto al mensaje en el texto del mecanismo de verificación.

Otros aspectos se han incluido en la generación del mecanismo de verificación para mejorar la seguridad del esquema. Además del texto compuesto por el mensaje y el OTP se ha incluido un sello de tiempo que contiene la fecha y la hora de la generación del mensaje de seguridad.

Dado que todas las técnicas aplicadas en la generación del mensaje de verificación se basan los retos CAPTCHA, podemos considerar nuestro mecanismo en si un CAPTCHA. A partir de este momento nos referiremos indiferentemente como CAPTCHA o mensaje de verificación.

4.1.2 Estudio de las soluciones de seguridad en el diseño

La seguridad del mecanismo diseñado se basa en la complejidad de llevar a cabo ciertas tareas por una máquina. En esta sección analizaremos los posibles ataques que se pueden realizar sobre el mecanismo diseñado y que medidas hemos tomado para evitar su éxito.

Suplantación del CAPTCHA

El ataque más lógico es el de suplantación de mensaje. El atacante captura el CAPTCHA, lo descarta y genera uno nuevo donde él puede elegir el texto enviado. El mecanismo de generación de mensajes está explícitamente diseñado para evitar este tipo de ataques y se ha añadido el secreto compartido como fondo del mensaje, por lo tanto la única forma de generarlo sin que el receptor pueda notar el cambio es conocer el concepto clave o bien tentar a la suerte y generar un mensaje con una imagen aleatoria. Asumiendo que el atacante conoce la lista de conceptos representados en la base de datos, sus posibilidades de realizar con éxito un CAPTCHA con un fondo aleatorio se verán limitadas por la longitud de la lista, es decir, a mayor número de conceptos menor probabilidad de que un ataque aleatorio sea eficaz. En la vida real, es muy difícil, por no decir imposible, que un elemento externo al sistema pueda listar las claves utilizadas ya que lo bueno de utilizar conceptos es que las posibilidades son ilimitadas. Además, el concepto tan sólo tiene que ser observado por el receptor sin que este deba aportar esta información en ningún momento del protocolo por lo que sería inútil realizar ataques en otros módulos del mismo para averiguarlo.

Extracción del secreto compartido

Es cierto que hoy en día no existe la tecnología suficiente para que una máquina pueda identificar el contenido de una imagen, más aún, si esta además contiene un texto sobreimpreso. Pero supongamos que es capaz de diferenciar el texto del background y que conoce las imágenes que son utilizadas en la generación del sistema de autenticación visual. Con estas premisas, es posible que con la tecnología actual se pudiera identificar cuál es la imagen, aunque sobre ella se hubieran realizado leves modificaciones como reducir la imagen mediante cortes en alguno de sus lados rescaldados o rotaciones, ya existen métodos de identificación de patrones de colores y formas por comparación muy avanzados. En este caso, y si el sistemas malicioso se dedicara a recopilar tickets y la base de datos de imágenes no es muy elevada, llegaría un momento en el que sistema podría analizar la relación entre los destinatarios y las imágenes descubriendo así el secreto entre ambas entidades.

Para evitar estos riesgos, hemos recuperado algunos estudios sobre las características de las imágenes y como afecta la modificación de ciertos parámetros, como son la luminancia, los colores o la aplicación de distintos filtros de difuminamiento y difusión, a la percepción de las personas y a los resultado de algunos de los métodos de comparación de imágenes más modernos. La universidad de Pennsylvania, en un estudio publicado en el 2009, demuestra que conociendo la base de datos, la modificación de un único tipo de parámetro debe de aplicarse de una forma muy agresiva para que no existan posibilidades de que sea reconocido por una máquina, lo cual afecta en gran medida a la percepción de las personas. Sin embargo, muestra resultados muy ventajosos para nuestro sistema cuando se aplican varias de estas transformaciones de forma leve. Por lo tanto, vemos muy ventajosa la aplicación de filtros que modifiquen de forma leve distintas características de la imagen sin llegar a ser está irreconocible para las personas.

Atacante humano

Dado que la seguridad del CAPTCHA se basa en un problema de la IA, es sencillo deducir que existe un fallo de seguridad al no tenerse en cuenta un ataque por parte de una persona. El único requisito para poder identificar la clave es tener capacidades humanas, es decir, cualquier persona que tenga acceso al CAPTCHA será capaz de ver la imagen e intuir cual es el concepto compartido, lo que no implica que su valoración sea correcta. Por lo tanto, una vez que se ha enviado una notificación donde se integre este mecanismo una máquina maliciosa podrá reenviárselo a una persona para que esta identifique la clave y así poder generar un nuevo mensaje suplantando la identidad del emisor original. Pero la persona puede tener dificultades identificando la clave ya que la imagen no es la clave en si sino la representación de un concepto u objeto, por lo que una misma imagen, como la imagen de la Figura 28, puede se identificativa de conceptos tan dispares como "felicidad" o "mujer joven y castaña". El sistema suplantador deber contar con una amplia base de datos de imágenes etiquetadas de forma muy precisa y tener la capacidad de adaptarse de forma efectiva a cualquiera de las descripciones ofrecidas por el humano que colabora con él.

Figura 28. Imágen con varios conceptos asociados

No se pretende que la eficacia del mecanismo se base en la robustez de la clave, esta acuerda previamente entre las partes comunicantes e independientemente sin ninguna intervención del sistema generador de CAPTCHAs.

El problema de un atacante en colaboración con una persona es punto clave en el protocolo y aún no se ha sido solucionado. Como medida adicional, aunque no como solución al problema, el background de la imagen estar compuesto por un mosaico de imágenes entre las que se encontrará la clave y un número de imágenes aleatorias, obligando al humano atacante a tener varios mensajes dirigidos a una misma persona para conocer la clave. El mecanismo de autenticación se integrará en un escenario de notificaciones en un único sentido, es decir, no permite que el receptor del mensaje realice peticiones de nuevas notificaciones por el mismo canal de comunicación que las recibe, las comunicaciones las inicia siempre el emisor, por lo tanto el tiempo que se debe esperar para tener varios instancias de mensajes dirigidos a un mismo receptor puede ser muy elevada y poco rentable. En este caso hemos decidido realizar un mosaico de dos imágenes, la clave y una segunda aleatoria, por lo que serán necesarias al menos dos instancias para que se pueda detectar la clave.

Modificación mensaje

Una vez comprendida la complejidad que supone para una máquina la generación de un nuevo CAPTCHA desde cero, un atacante podría simplemente tratar de insertar, borrar o modificar el texto transmitido dentro de la imagen. La máquina nos sólo debe identificar el texto y localizar los límites exactos de cada uno de los caracteres, sino que además debe ser capaz de rellenar el espacio que estos ocupaban con nuevo texto, ya que simular partes de la imagen es una tecnología que todavía no se ha desarrollado.

Sin lugar a dudas hemos centrado la robustez del mecanismo en que el mensaje no pueda ser localizado e identificado para ello nos hemos basado en las técnicas de implementación de CAPTCHAs basados en texto. El texto enviado es un código lo que evita que se puedan hacer ataques al diccionario, además el sistema cuenta con más de 1000 tipos de letra donde los colores utilizados para la escritura de caracteres aunque para el ojo humano aparentes ser negros son colores elegidos en la gama de los grises que favorece su inserción en el texto pudiendo pasar por zonas oscuras de la imagen y se han utilizado técnicas para evitar ataques típicos como la aplicación de OCRs sobre las imágenes para detectar texto, en la actualidad el algoritmo más peligros es el de segmentación de caracteres.

Por lo tanto, las probabilidades de que una modificación del texto no sea detectada por el receptor son muy bajas.

Duplicación del CAPTCHA

Es posible que el atacante decida almacenar uno de los mensajes, para posteriormente enviarlo como muestra de autenticación en otro mensaje o con cualquier otro fin malicioso. En ese caso y para evitar ataques por duplicación se incluye dentro del CAPTCHA un sello de tiempo que indica la hora y fecha en la que se genero para que el propio receptor pueda notar el desfase de tiempo y desconfíe de la notificación.

4.1.3 Esquema físico de la solución

Dado que el proceso necesario para poder generar un mensaje de verificación es muy complejo para llevarse a cabo por una tarjeta inteligente, donde los recursos de memoria y de ejecución son limitados, es necesario añadir un nuevo elemento al sistema. Este elemento consiste en un servidor que será capaz de procesar peticiones de generación de CAPTCHAs. Por lo tanto el conjunto de entidades que componen el sistema es el que se muestra en la **Figura** 29.

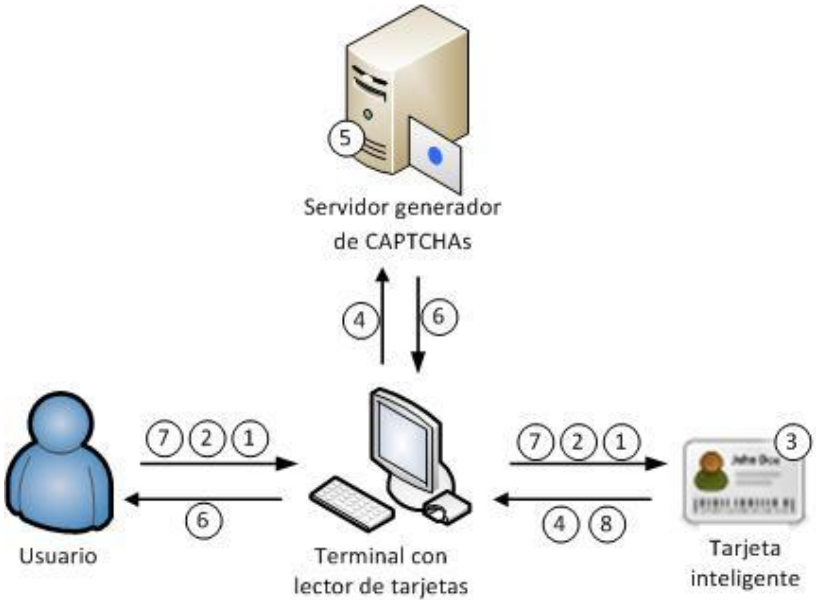

Figura 29. Esquema físico de la solución

El protocolo seguido para realizar la firma digital es el siguiente:

9. **Acceso al sistema.** El usuario introducirá su código PIN en la aplicación del terminal, este enviará la instrucción de verificación de número PIN a la tarjeta inteligente donde se verificará.
10. **Inserción del mensaje.** Si la tarjeta inteligente confirma que el número PIN introducido en el paso 1 es correcto, el terminal pide mediante un mensaje por pantalla que el usuario introduzca el mensaje que desea firmar. El terminal recoge el mensaje y se lo transmite a la tarjeta inteligente.
11. **Petición de CAPTCHA.** Cuando la tarjeta inteligente recibe un mensaje, lo almacena y genera una petición de CAPTCHA para enviar al servidor generador de CAPTCHAS. La petición está compuesta por tres campos:
    - Mensaje a firmar enviado por el usuario + OTP cifrado por la clave pública del servidor.
    - Firma digital del mensaje cifrado.
    - Identificador de usuario.
12. **Envió de la petición.** La tarjeta transmite al terminal la petición y este se encargará de enviarla hasta el servidor de CAPTCHAS a través de una conexión.
13. **Generación CAPTCHA.** El servidor comprueba que conoce al usuario y que tiene almacenada su clave pública. Comprueba la integridad del mensaje mediante la firma, si es capaz de verificarlo, entonces, descifrado el mensaje y procede a generar un CAPTCHA que contenga el mensaje y el OTP con la clave asociada al id del usuario incluido en la petición.
14. **Retorno de CAPTCHA.** El CAPTCHA se envía al terminal mediante la misma conexión iniciada en el paso 4. El terminal procede a mostrarlo.
15. **Confirmación del mensaje.** Si el mensaje es autentico el usuario introduce la clave OTP en el terminal, que deberá transmitirla hasta la tarjea.
16. **Firmar mensaje.** Cuando la tarjeta inteligente recibe un OTP correcto firma el mensaje que había almacenado en el paso 2 y se la envía junto con el mensaje al terminal.

A modo sintético e ilustrativo, se muestra el modelo de la solución y el funcionamiento del sistema mediante un diagrama de secuencia que describe los pasos necesarios para realizar un proceso de firma.

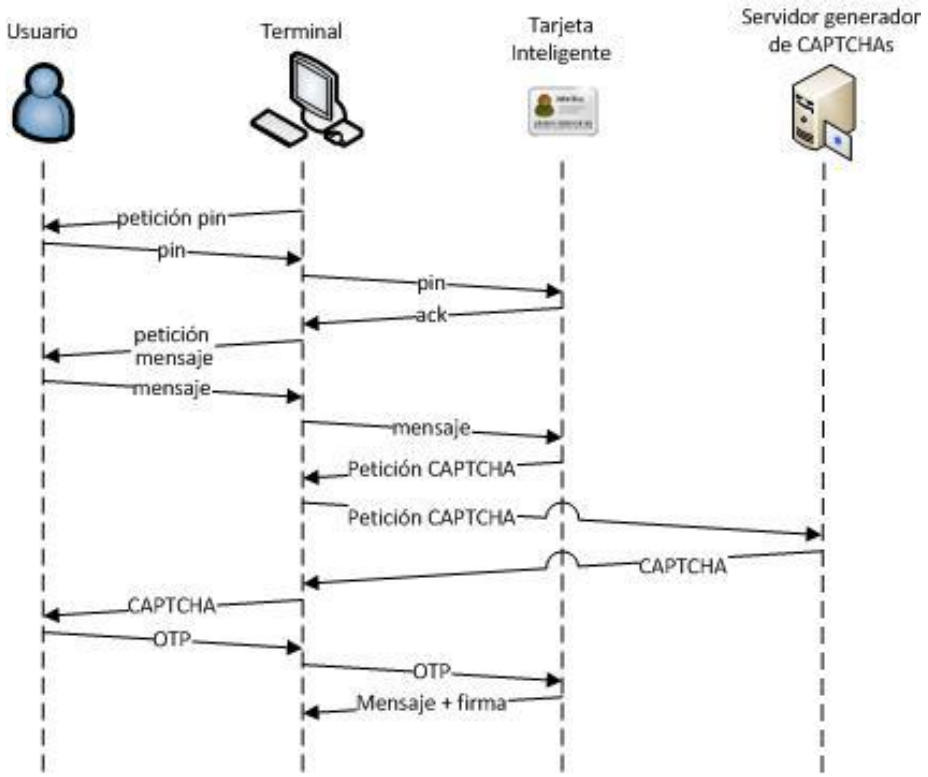

Figura 30. Diagrama secuencial de las comunicaciones

4.2 Requisitos software

En este apartado se definen los requisitos software de la solución a desarrollar. La clasificación de los mismos se ha realizado siguiendo las recomendaciones de la ESA3, que identifica los siguientes tipos de requisitos:

- **Requisitos funcionales:** Los requisitos funcionales especifican aquellas funciones que el sistema, o un componente del mismo, debe ser capaz de realizar.

- **Requisitos de rendimiento:** Los requisitos de restricción especifican valores numéricos para variables medibles utilizadas para definir una función (por ejemplo, ratio frecuencia, capacidad, velocidad y fiabilidad).

- **Requisitos de interfaz:** Los requisitos de interfaz especifican el hardware, software o elementos de bases de datos del sistema, o de un componente del sistema, que deben interactuar con la solución a desarrollar. Estos requisitos se clasificarán en: interfaces software, hardware o comunicación. Adicionalmente, se establecerá una clasificación que diferencie entre interfaces externos o internos, dependiendo si coincide o no con los límites del sistema.

[3] Guide to applying the ESA Software Engineering Standards to small software projects, BSSC (96)2.

- **Requisitos operacionales:** Los requisitos operacionales especifican cómo funcionará el sistema y cómo se comunicará con los operadores humanos (por ejemplo: pantalla, teclado, etc.). Además estos requisitos pueden describir aspectos físicos del interfaz de usuario (descripciones de los diálogos, lenguaje de comandos, diseños de pantallas, etc.).

- **Requisitos de recursos:** Los requisitos de recursos especifican los límites superiores de los recursos físicos, tales como la capacidad de cómputo, memoria, espacio en disco, etc.

- **Requisitos de verificación:** Los requisitos de verificación restringen el diseño de la solución. Lo hacen exigiendo las características que facilitan la verificación de las funciones del sistema, de forma que puedan detectar y eliminar defectos de implementación.

- **Requisitos de aceptación:** Los requisitos de verificación restringen el diseño de la solución. Deben comprobar que la funcionalidad ofrecida por el sistema es la deseada por el cliente.

- **Requisitos de documentación:** Los requisitos de documentación describen aquellos requisitos relativos al estilo y formato de los documentos generados en el proyecto.

- **Requisitos de seguridad:** Los requisitos de seguridad especifican las medidas que adopta el sistema para controlar las amenazas sobre la confidencialidad, integridad y disponibilidad. Deben describir el nivel y frecuencia de acceso permitido a los usuarios autorizados. Además debe ser descrito el tipo de usuarios sin autorización de acceso.

- **Requisitos de portabilidad:** Los requisitos de portabilidad especifican la facilidad para mover el software de un entorno operacional a otro. Los equipos o sistemas operativos que puedan alojar el sistema deben ser declarados. Estos requisitos pueden reducir el rendimiento de la solución y aumentar el esfuerzo para desarrollarla, por lo que deben ser declarados considerando el tiempo de vida estimada, el sistema operativo y el hardware de la solución final.

- **Requisitos de calidad:** Los requisitos de calidad especifican los atributos del software que lo hacen apropiados para su propósito. Los principales atributos de calidad sobre fiabilidad, mantenimiento y seguridad de personas deben definirse de forma separada.

- **Requisitos de fiabilidad:** Los requisitos de fiabilidad especifican la capacidad del sistema, o componente, para realizar las funciones requeridas, bajo unas determinadas condiciones y un periodo de tiempo específico. Para medir la fiabilidad de acuerdo a la definición anterior se utiliza la métrica “Mean Time Between Failure” (MTBF). La especificación de este tipo de requisitos debe proporcionar una clasificación de fallos basada en su severidad.

Para cada uno de los requisitos definidos en las siguientes subsecciones, se han recogido en una tabla donde se recogen los siguientes datos:

1. **Identificador del Requisito:** Identificador único del requisito que se está describiendo. La codificación de identificadores utilizada ha sido la siguiente:
    - RS – CC – XXX, donde:
        - RS: Representa un requisito software

- CC: Código de identificación del tipo de requisito software.
- XXX: Número de secuencia del requisito. Se corresponde con un número entre 001 y 999, siendo 001 el primer número asignado a un requisito y que crecerá de manera secuencial.

Código (CC)	Significado
FUN	Requisitos Funcionales
REN	Requisitos de Rendimiento
INT	Requisitos de Interfaz
OPE	Requisitos Operacionales
REC	Requisitos de Recursos
VER	Requisitos de Verificación
PRU	Requisitos de Pruebas de Aceptación
DOC	Requisitos de Documentación
SEG	Requisitos de Seguridad
POR	Requisitos de Portabilidad
CAL	Requisitos de Calidad
FIA	Requisitos de Fiabilidad

Tabla 19. Códigos de requisitos

2. **Dependencias:** Identificadores de otros requisitos con los que el requisito descrito guarda algún tipo de relación.
3. **Nombre:** Nombre específico del requisito descrito**.**
4. **Descripción:** Representa una descripción textual de la naturaleza y objetivo del requisito. La descripción debe ser breve, concisa y sin ambigüedad.
5. **Prioridad:** Medida de la prioridad del requisito descrito con el fin de establecer una estrategia de desarrollo. Determina la importancia de cumplir con el requisito antes de abordar otros requisitos o actividades productivas. Su valor puede ser:
    - **Alta:** Máxima prioridad en abordar el cumplimiento del requisito.

- **Media:** El requisito es importante, pero no es el más prioritario.
- **Baja:** El requisito no es prioritario.

6. **Fuente:** Especifica el requisito de usuario que origina el requisito descrito.
7. **Necesidad:** La necesidad marca cómo de importante y útil es el requisito dentro del sistema y su influencia en la plena satisfacción del cliente. La necesidad de un requisito puede ser:
   - **Esencial:** El requisito ha de ser cumplido obligatoriamente.
   - **Deseable:** El requisito no es obligatorio, pero aportaría mayor calidad y satisfacción al cliente.
   - **Opcional:** El requisito puede no ser cumplido sin influir negativamente en la satisfacción final.
8. **Estabilidad:** Indica cómo de probable es que el requisito pueda ser modificado en el futuro, o incluso eliminado. Los valores de este atributo pueden ser:
   - **Alta:** El requisito no va a ser nunca modificado.
   - **Media:** El requisito puede ser modificado en algún momento del desarrollo.
   - **Baja:** El requisito será modificado con una alta probabilidad durante el desarrollo.

4.2.1 Requisitos funcionales

Identificador	RS-FUN-001	**Dependencias**	No aplica
Nombre	Identificación del usuario		
Descripción	El sistema identificará unívocamente al usuario.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUC-001
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 20. RS-FUN-001

Identificador	RS-FUN-002	**Dependencias**	No aplica
Nombre	Bloqueo sistema		
Descripción	El sistema se bloqueará si se realizan 3 intentos de autenticación de usuario fallidos.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUR-004
Necesidad	☑ Esencial ☐ Deseable	**Estabilidad**	☑ Alta ☐ Media

	☐ Opcional		☐ Baja

Tabla 21. RS-FUN-002

Identificador	RS-FUN-003	**Dependencias**	No aplica
Nombre	Introducir mensaje		
Descripción	El sistema aceptará mensajes introducidos por el usuario.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUC-002
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 22. RS-FUN-003

Identificador	RS-FUN-004	**Dependencias**	No aplica
Nombre	1.1.1 Generación OTP		
Descripción	El sistema al leer un mensaje introducido por el usuario generará, dentro de la tarjeta inteligente, un número aleatorio que hará la función de OTP de confirmación.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUR-008
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 23. RS-FUN-004

Identificador	RS-FUN-005	**Dependencias**	RS-FUN-003
Nombre	Generación de petición de CAPTCHAs		
Descripción	El sistema al leer un mensaje introducido por el usuario construirá una petición de generación de CAPTCHA.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUC-003
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 24. RS-FUN-005

Identificador	RS-FUN-006	**Dependencias**	No aplicable
Nombre	Generación texto CAPTCHA		
Descripción	El sistema cifrará el mensaje que el usuario quiere firmar, el OTP de confirmación y la fecha y hora de la encriptación mediante las técnicas desarrolladas para la generación de CAPTCHAs basados en texto.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUC-003
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 25. RS-FUN-006

Identificador	RS-FUN-007	**Dependencias**	No aplicable
Nombre	Generación imagen CAPTCHA		
Descripción	El sistema se autenticará como generador del CAPTCHA añadiendo como fondo de este una imagen representativa del concepto acordado entre el sistema y el usuario como clave compartida.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUR-007
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 26. RS-FUN-007

Identificador	RS-FUN-008	**Dependencias**	No aplicable
Nombre	Aplicación de ruido en CAPTCHA		
Descripción	El sistema añadirá al fondo del CAPTCHA una imagen aleatoria que no aportará información.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Desarrollador
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 27. RS-FUN-008

Identificador	RS-FUN-009	Dependencias	No aplicable
Nombre	Mostrar CAPTCHA		
Descripción	El sistema mostrará el CAPTCHA al usuario		
Prioridad	☑ Alta ☐ Media ☐ Baja	Fuente	RUC-003
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	Estabilidad	☑ Alta ☐ Media ☐ Baja

Tabla 28. RS-FUN-009

Identificador	RS-FUN-010	Dependencias	No aplicable
Nombre	Verificación operación de firma		
Descripción	El sistema leerá por tecado el OTP confirmación de firma de mensaje.		
Prioridad	☑ Alta ☐ Media ☐ Baja	Fuente	RUC-003
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	Estabilidad	☑ Alta ☐ Media ☐ Baja

Tabla 29. RS-FUN-010

Identificador	RS-FUN-011	Dependencias	No aplicable
Nombre	Invalidación petición de firma		
Descripción	Se considerará no válida la petición de firma si se introducen 3 confirmaciones no válidas y el sistema se cerrará.		
Prioridad	☑ Alta ☐ Media ☐ Baja	Fuente	RUC-003
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	Estabilidad	☑ Alta ☐ Media ☐ Baja

Tabla 30. RS-FUN-011

Identificador	RS-FUN-012	Dependencias	No aplicable
Nombre	Fichero con firma del mensaje		

Descripción	El sistema generará un fichero que contiene el mensaje y la firma.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUC-002
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 31. RS-FUN-012

Identificador	RS-FUN-013	**Dependencias**	No aplicable
Nombre	Cierre sistema		
Descripción	El sistema se cerrará al finalizar el proceso de firma		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUC-004
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 32. RS-FUN-013

Identificador	RS-FUN-014	**Dependencias**	No aplicable
Nombre	Cierre sistema al retirar la tarjeta inteligente		
Descripción	El sistema interrumpirá el proceso de firma y se cerrará si se retira la tarjeta inteligente del lector.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUC-005
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 33. RS-FUN-014

Identificador	RS-FUN-015	**Dependencias**	No aplicable
Nombre	Cierre sistema por usuario		
Descripción	El sistema interrumpirá el proceso de firma y se cerrará si en cualquier punto del protocolo el usuario introduce la secuencia de finalización.		
Prioridad	☑ Alta ☐ Media	**Fuente**	RUC-009

	☐ Baja		
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 34. RS-FUN-015

4.2.2 Requisitos de rendimiento

Identificador	RS-REN-001	**Dependencias**	No aplicable
Nombre	Tiempo de generación de CAPTCHA		
Descripción	El proceso de generación de CAPTCHAs no debe durar más de 1s.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Desarrollador
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 35. RS-REN-001

Identificador	RS-REN-002	**Dependencias**	No aplicable
Nombre	Tamaño CAPTCHA		
Descripción	El mensaje de confirmación o CAPTCHA no debe ocupar más de 1MB		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Desarrollador
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 36. RS-REN-002

4.2.3 Requisitos de interfaz

Identificador	RS-INT-001	**Dependencias**	No aplicable
Nombre	Tarjeta JavaCard		
Descripción	La tarjeta inteligente utilizada debe ser una JavaCard		

Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Desarrollador
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 37. RS-INT-001

Identificador	RS-INT-002	**Dependencias**	No aplicable
Nombre	Formato tarjeta inteligente		
Descripción	Es sistema sólo aceptará tarjetas inteligentes que contengan los siguientes datos: • Par de claves RSA del usuario • Identificador del usuario • Clave pública del servidor de generación de claves		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUR-005
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 38. RS-INT-002

Identificador	RS-INT-003	**Dependencias**	No aplicable
Nombre	Comunicación con usuario		
Descripción	El sistema se comunicará con el usuario mediante una pantalla y un teclado		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUR-009
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 39. RS-INT-003

Identificador	RS-INT-004	**Dependencias**	No aplicable
Nombre	Comunicación con tarjeta inteligente		
Descripción	El sistema se comunicará con la tarjeta inteligente mediante un lector de tarjetas		

Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUR-011
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 40. RS-INT-004

Identificador	RS-INT-005	**Dependencias**	No aplicable
Nombre	Comunicación con servidor		
Descripción	Las comunicaciones entre el terminal y el servidor del sistema se realizarán mediante conexiones segura SSL con autenticación del servidor.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUC-001
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 41. RS-INT-005

4.2.4 Requisitos operacionales

Identificador	RS-OPE-001	**Dependencias**	RS-FUN-001
Nombre	Identificación con tarjeta Inteligente		
Descripción	El sistema realizará la identificación del usuario en la tarjeta inteligente.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUR-002
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 42. RS-OPE-001

Identificador	RS-OPE-002	**Dependencias**	RS-FUN-005
Nombre	Petición de generación de CAPTCHA		
Descripción	La petición de generación de CAPTCHA contendrá la siguientes información: • Identificador de usuario que realiza la petición • Mensaje encriptado por la clave pública del servidor		

	• Firma digital del mensaje encriptado realizada por el usuario		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUC-003
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 43. RS-OPE-002

Identificador	RS-OPE-003	**Dependencias**	RS-SEG-004
Nombre	Cifrado por tarjeta inteligente		
Descripción	El sistema cifrará el mensaje introducido por el usuario junto con el OTP asociado dentro de la tarjeta inteligente.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUR-008
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 44. RS-OPE-003

Identificador	RS-OPE-004	**Dependencias**	No aplicable
Nombre	Firma de petición en tarjeta inteligente		
Descripción	El sistema realizará la firma digital de la petición de generación de CAPTCHAs dentro de la tarjeta inteligente.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUR-008
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 45. RS-OPE-004

Identificador	RS-OPE-005	**Dependencias**	No aplicable
Nombre	Uso del sistema		
Descripción	El sistema sólo permitirá que un usuario realice un proceso firma de simultáneamente.		
Prioridad	☑ Alta ☐ Media	**Fuente**	Desarrollador

	☐ Baja		
Necesidad	☐ Esencial ☑ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 46. RS-OPE-005

Identificador	RS-OPE-006	**Dependencias**	No aplicable
Nombre	Verificación de OTP en tarjeta inteligente		
Descripción	La verificación del OTP se realizará dentro de la tarjeta.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Desarrollador
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 47. RS-OPE-006

Identificador	RS-OPE-007	**Dependencias**	No aplicable
Nombre	Firma del mensaje en la tarjeta inteligente		
Descripción	La firma del mensaje se realizará dentro de la tarjeta.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUC-002
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 48. RS-OPE-007

4.2.5 Requisitos de recursos

Identificador	RS-REC-001	**Dependencias**	No aplicable
Nombre	Claves RSA usuario		
Descripción	La tarjeta inteligente tendrá almacenado previamente el par de claves RSA del usuario.		
Prioridad	☑ Alta	**Fuente**	RUR-005

	☐ Media ☐ Baja		
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 49. RS-REC-001

Identificador	RS-REC-002	**Dependencias**	No aplicable
Nombre	Clave pública servidor		
Descripción	La tarjeta inteligente tendrá almacenada previamente la clave pública del servidor.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUR-005
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 50. RS-REC-002

Identificador	RS-REC-003	**Dependencias**	No aplicable
Nombre	Identificador usuario		
Descripción	La tarjeta inteligente tendrá almacenado previamente el identificador del usuario.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUR-006
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 51. RS-REC-003

Identificador	RS-REC-004	**Dependencias**	No aplicable
Nombre	Datos servidor		
Descripción	El servidor del sistema tendrá almacenado previamente el número identificador del usuario y la clave pública RSA que le identifica.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Desarrollador
Necesidad	☑ Esencial ☐ Deseable	**Estabilidad**	☑ Alta ☐ Media

	☐ Opcional		☐ Baja

Tabla 52. RS-REC-004

4.2.6 Requisitos de verificación

Identificador	RS-VER-001	**Dependencias**	No aplicable
Nombre	Existencia de usuario		
Descripción	El servidor generador de CAPTCHAs verificará que el identificador de usuario adjunto en la petición de generación de CAPTCHAS corresponde a un usuario registrado en el sistema.		
Prioridad	☐ Alta ☑ Media ☐ Baja	**Fuente**	RUR-006
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 53. RS-VER-001

Identificador	RS-VER-002	**Dependencias**	No aplicable
Nombre	Verificación de firma		
Descripción	El servidor generador de CAPTCHAs verificará que la firma del mensaje cifrado pertenece al usuario adjuntado en al petición de generación de CAPTCHA.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Desarrollador
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 54. RS-VER-002

Identificador	RS-VER-003	**Dependencias**	No aplicable
Nombre	Verificación para descifrado de mensaje		
Descripción	El servidor no descifrará el mensaje de la petición hasta que se verifique la firma digital de este es correcta.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Desarrollador

Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 55. RS-VER-003

Identificador	RS-VER-004	**Dependencias**	No aplicable
Nombre	Verificación OTP		
Descripción	El sistema no firmará el mensaje del usuario hasta que este confirme la acción introduciendo del OTP.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Desarrollador
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 56. RS-VER-004

4.2.7 Requisitos de documentación

Identificador	RS-DOC-001	**Dependencias**	RS-FUN-005
Nombre	Fichero de petición de CAPTCHA		
Descripción	La petición de generación de CAPTCHA será un fichero XML que debe cumplir el siguiente esquema DTD. <?xml version="1.0" encoding="UTF-8"?> <!-- DTD CAPTCHArequest --> <!ELEMENT CAPTCHArequest (message, signature, user) > <!ELEMENT message (#PCDATA)> <!ELEMENT signature (#PCDATA)> <!ELEMENT user (#PCDATA)>		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Desarrollador
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 57. RS-DOC-001

4.2.8 Requisitos de seguridad

Identificador	RS-SEG-001	Dependencias	No aplica
Nombre	Código PIN		
Descripción	El PIN estará formado por 5 dígitos.		
Prioridad	☑ Alta ☐ Media ☐ Baja	Fuente	RUR-004
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	Estabilidad	☑ Alta ☐ Media ☐ Baja

Tabla 58. RS-SEG-001

Identificador	RS-SEG-002	Dependencias	RS-FUN-004
Nombre	Asociación mensaje OTP		
Descripción	Cada mensaje tendrá asociado una OTP de confirmación.		
Prioridad	☑ Alta ☐ Media ☐ Baja	Fuente	RUR-008
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	Estabilidad	☑ Alta ☐ Media ☐ Baja

Tabla 59. RS-SEG-002

Identificador	RS-SEG-003	Dependencias	RS-FUN-004
Nombre	Clave OTP		
Descripción	La OTP de confirmación de firma de mensaje estar compuesta por 4 dígitos		
Prioridad	☑ Alta ☐ Media ☐ Baja	Fuente	RUC-008
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	Estabilidad	☑ Alta ☐ Media ☐ Baja

Tabla 60. RS-SEG-003

Identificador	RS-SEG-004	Dependencias	No aplicable
Nombre	Cifrado de mensaje		

Descripción	El mensaje de la petición de generación de CAPTCHA estará cifrado con la clave pública del servidor del sistema		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	RUC-008
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 61. RS-SEG-004

Identificador	RS-SEG-005	**Dependencias**	No aplicable
Nombre	Función de firma		
Descripción	El sistema realizará las firmas digitales mediante la función resumen SHA1 y la clave privada RSA del usuario.		
Prioridad	☑ Alta ☐ Media ☐ Baja	**Fuente**	Desarrollador
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 62. RS-SEG-005

4.2.9 Requisitos de portabilidad

Identificador	RS-POR-001	**Dependencias**	No aplica
Nombre	Solución portable		
Descripción	El sistema debe ser una aplicación multiplataforma.		
Prioridad	☐ Alta ☑ Media ☐ Baja	**Fuente**	Desarrollador
Necesidad	☐ Esencial ☑ Deseable ☐ Opcional	**Estabilidad**	☑ Alta ☐ Media ☐ Baja

Tabla 63. RS-POR-001

Identificador	RS-POR-002	Dependencias	No aplica
Nombre	Tarjetas Inteligentes Adaptables a la solución		
Descripción	La solución será adaptable a otras tarjetas inteligentes que cumplan los estándares PKCS#11.		
Prioridad	☐ Alta ☑ Media ☐ Baja	Fuente	Desarrollador
Necesidad	☐ Esencial ☑ Deseable ☐ Opcional	Estabilidad	☑ Alta ☐ Media ☐ Baja

Tabla 64. RS-POR-002

4.2.10 Requisitos de calidad

Identificador	RS-CAL-001	Dependencias	No aplicable
Nombre	Legibilidad del texto del CAPTCHA		
Descripción	El texto incluido en el CAPTCHA tiene que ser legible para una persona.		
Prioridad	☑ Alta ☐ Media ☐ Baja	Fuente	Desarrollador
Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	Estabilidad	☑ Alta ☐ Media ☐ Baja

Tabla 65. RS-CAL-001

Identificador	RS-CAL-002	Dependencias	No aplicable
Nombre	Identificación de la clave en el CAPTCHA		
Descripción	Las imágenes incluidas en el CAPTCHA tienen que ser interpretables por una persona.		
Prioridad	☑ Alta ☐ Media ☐ Baja	Fuente	Desarrollador
Necesidad	☑ Esencial ☐ Deseable	Estabilidad	☑ Alta ☐ Media

	☐ Opcional		☐ Baja

Tabla 66. RS-CAL-002

4.3 Matriz de trazabilidad

En la tabla X se muestra la correspondencia entre los requisitos de usuario y los requisitos software que los resuelven mediante una matriz de trazabilidad, de esta manera se puede comprobar que se están cumpliendo las especificaciones recogidas en el capítulo 2 Requisitos de usuario.

	RUC-001	RUC-002	RUC-003	RUC-004	RUC-005	RUR-001	RUR-002	RUR-003	RUR-004	RUR-005	RUR-006	RUR-007	RUR-008	RUR-009	RUR-010	RUR-011
RS-FUN-001	X															
RS-FUN-002									X							
RS-FUN-003		X														
RS-FUN-004													X			
RS-FUN-005			X													
RS-FUN-006			X													
RS-FUN-007												X				
RS-FUN-008																
RS-FUN-009			X													
RS-FUN-010			X													
RS-FUN-011			X													
RS-FUN-012		X														
RS-FUN-013				X												
RS-FUN-014					X											
RS-FUN-015														X		
RS-REN-001																
RS-REN-002																
RS-INT-001																
RS-INT-002										X						
RS-INT-003														X		
RS-INT-004																X
RS-INT-005						X										
RS-OPE-001							X									
RS-OPE-002			X													
RS-OPE-003													X			

RS-OPE-004																
RS-OPE-005																
RS-OPE-006																
RS-OPE-007		X														
RS-REC-001										X						
RS-REC-002										X						
RS-REC-003										X						
RS-REC-004																
RS-VER-001											X					
RS-VER-002																
RS-VER-003																
RS-VER-004																
RS-DOC-001																
RS-SEG-001									X							
RS-SEG-002													X			
RS-SEG-003													X			
RS-SEG-004													X			
RS-SEG-005																
RS-POR-001																
RS-POR-002																
RS-CAL-001																
RS-CAL-002																

Tabla 67. Matriz de trazabilidad

Capítulo 5

DESCRIPCIÓN E IMPLEMENTACIÓN DE LOS COMPONENTES

5.1 Descripción de la descomposición

Para realizar la descripción de la descomposición de la solución, se han identificado los componentes, o entidades, que son relevantes en el proceso de firma digital y que deben tenerse en cuenta para comprender el comportamiento del diseñado.

En el diagrama de componentes, se han descrito tanto los propios componentes que forman parte de la solución, como las dependencias de uso entre ellos (identificadas mediante flechas discontinuas, la punta de flecha señala el componente independiente de la relación). Así mismo, se han identificado mediante el símbolo Interfaz ○—— las interfaces implementadas por cada componente para realiza las comunicaciones necesarias.

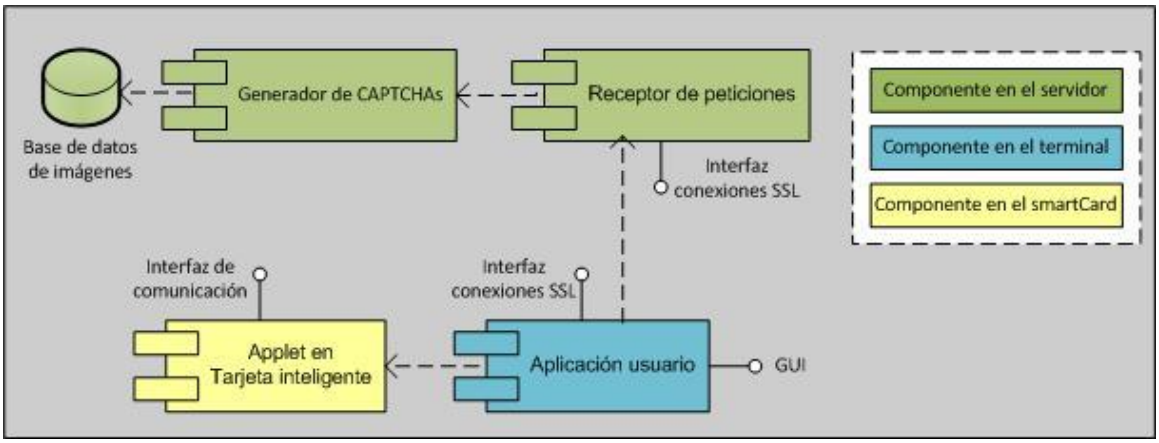

Figura 31. Diagrama de componentes

5.2 Descripción de componentes

5.2.1 Componente de terminal

5.2.1.1 Arquitectura

La aplicación desarrollada en la plataforma del terminal se pude considerar el controlador de protocolo diseñado en el capítulo anterior. Mediante este componente se integran los tres participantes del protocolo: usuario, tarjeta inteligente y servidor de CAPTCHAs. Todas las comunicaciones establecidas entre ellos son gestionadas en este componente.

Este componente es el que inicia las comunicaciones y se ha diseñado como cliente del resto de componentes enla arquitectura global de la solución, es decir, se conecta con el resto de componentes para realizar peticiones de los datos que él no es capaz de elaborar.

Tiene una estructura muy sencilla que consta: de una clase principal, ProgramaUsuario, que describe las funcionalidades del usuario; y de otras tres secundarias que actúan como interfaces para realizar las comunicaciones con las entidades participantes en el sistema. La clase GUI, es la interfaz gráfica de entrada y salida de datos del usuario; la clase BufferTarjeta, es la conexión con la tarjeta inteligente adaptadas a la interfaz de esta; y la clase SSLConexiónCliente, gestiona una conexión SSL con el servidor del sistema.

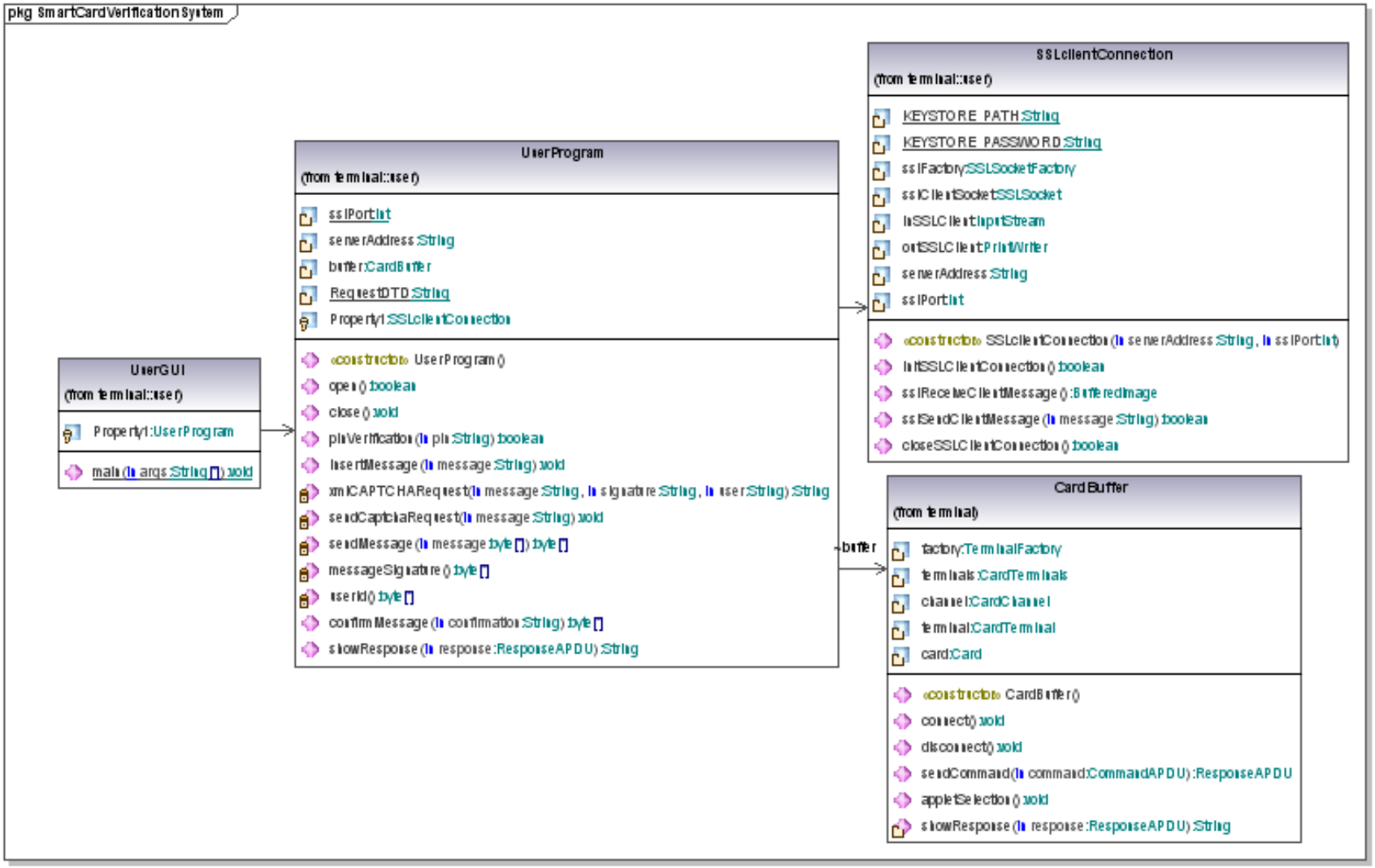

Figura 32. Diagrama de clases componente en termianl

5.2.1.2 Componentes

Identificador	1. Terminal	**Tipo**	Subcomponente
Funcionalidad	Aplicación en el terminal.		
Interfaces	Interactúa directamente con el usuario, realiza conexiones SSL con el servidor y crea un canal de comunicación con la tarjeta inteligente.		
Procesamiento	Gestiona las comunicaciones entre el usuario, la tarjeta inteligente y el servidor de generación de CAPTCHAs.		
Datos	Controla la secuencia de los procesos para generar la firma digital según los pasos definidos por el protocolo de seguridad diseñado.		
Manejo de Errores	Recoge los errores que surgen durante las comunicaciones y lo redirige a las entidades implicadas.		
Dependencias	Ninguna.		

Tabla 68. Terminal

Identificador	1.1 BufferTarjeta	**Tipo**	Clase
Funcionalidad	Implementa la interfaz de comunicación con la tarjeta inteligente.		
Interfaces	**Entrada:** instrucciones para la tarjeta inteligente. **Salida:** código de respuesta de la tarjeta inteligente.		
Procesamiento	Realiza la conexión con la tarjeta inteligente y transforma las peticiones en instrucciones legibles para la tarjeta inteligente.		
Datos	Los datos son adaptados de lenguaje Java a lenguaje comprensible para la tarjeta inteligente.		
Manejo de Errores	Informa a la aplicación del usuario de los problemas de conexión con la tarjeta inteligente.		
Dependencias	Tarjeta inteligente.		

Tabla 69. bufferTarjeta

Identificador	1.2 ConexiónClienteSSL	**Tipo**	Clase
Funcionalidad	Implementa la interfaz de comunicación con el servidor.		
Interfaces	**Entrada:** entrada de datos conexión SSL. **Salida:** salida de datos conexión SSL.		
Procesamiento	Construye una socket seguro mediante el protocolo SSL con el servidor.		
Datos	Todos los datos transmitidos a través de transmisión estarán provistos de integridad, confidencialidad y autenticación en el receptor.		
Manejo de Errores	Informa a la aplicación del usuario de los problemas de conexión con la el servidor.		
Dependencias	Servidor.		

Tabla 70. ConexiónClienteSSL

Identificador	1.3 GUI	**Tipo**	Clase
Funcionalidad	Implementa la interfaz grafica del usuario.		

Interfaces	**Entrada:** datos introducido por el usuario. **Salida:** respuestas del sistema.
Procesamiento	Es la entidad que comunica al usuario y el resto del sistema.
Datos	Recibe los datos directamente del usuario por la entrada estándar y muestra todas las comunicaciones del sistema por la salida estándar.
Manejo de Errores	Muestra mensajes de error compresibles para el usuario indicándole el estado del sistema.
Dependencias	Programa usuario.

Tabla 71. GUI

Identificador	1.4 ProgramaUsuario	**Tipo**	Clase
Funcionalidad	Implementa las capacidades del usuario en el sistema.		
Interfaces	Ninguna.		
Procesamiento	Realiza el control secuencial de las interacciones entre los componentes del sistema.		
Datos	Trata los datos que manejas cada uno de los componentes del sistema y direcciona hasta sus destinatarios.		
Manejo de Errores	Recoge los errores durante la ejecución del sistema y se los muestra al usuario.		
Dependencias	Interfaces de conexión a la tarjeta inteligente y al servidor.		

Tabla 72. ProgramaUsuario

5.2.1.3 Algoritmo

El orden de las funcionalidades está marcado por la interfaz gráfica del usuario que inicializa todos los procesos:

1. **Acceso al sistema.** Se le pide al usuario el código PIN, para su comprobación se conecta con la tarjeta inteligente y que será quien lo compruebe.

2. **Introducir mensaje.** Se insta al usuario para que introduzca el mensaje que desea firmar. Esta entrada inicia la funcionalidad de generación de petición de CAPTCHA que consiste en generar un mensaje XML con la siguiente estructura:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- DTD CAPTCHArequest -->
<!ELEMENT CAPTCHArequest (message, signature, user)>
   <!ELEMENT message (#PCDATA)>
   <!ELEMENT signature (#PCDATA)>
   <!ELEMENT user (#PCDATA)>
```

Cada uno de los campos se rellenaran realizando peticiones a la tarjeta inteligente, la primera conexión consistirá en el envío del mensaje, que le devolverá el propio mensaje con el OTP

encriptado. El segundo será la petición de la firma digital del mensaje anterior y por último se le requerirá a la tarjeta en número de usuario que tiene asociado.
Mediante una conexión SSL se enviará la petición de generación de CAPTCHAs al servidor que deberá responder con la imagen ha construido. La aplicación mostrará la imagen por pantalla.

3. **Verificación.** Se espera que el usuario inserte el código de verificación que se enviara por la conexión a la tarjeta inteligente, está indicará si es correcto devolviendo la firma del mensaje.

4. **Fin sistema.** Se cierran todas las conexiones se vuelve al estado inicial, paso 1.

5.2.2 Componente de servidor

En la implementación de la plataforma donde se encuentra el servidor se pueden distinguir una diferenciación clara entre las funcionalidades básicas de un servidor como tal, como son la gestión de conexión, recepción de mensaje, etc. y las propias del módulo de generación del mecanismo para el envío de mensajes seguros. Se ha creído conveniente separar estos ámbitos para hacer más reutilizable las solución de seguridad diseñada y poderla integrar con facilidad en cualquier otro escenario.

5.2.2.1 Subcomponente receptor de peticiones

5.2.2.1.1 Arquitectura

La arquitectura es muy sencilla, básicamente, se trata de una única entidad que recibe las peticiones mediante una conexión SSL en la que actúa como servidor e interpreta el contenido de la petición. Se ayuda de dos clases secundarias: SSSLserverConection, que gestiona la conexión SSL y peticiónManager que actúa como intérprete de la estructura XML de la petición.

5.2.2.1.2 Componentes

Identificador	2.2.1 SecurityServerMain	**Tipo**	Clase
Funcionalidad	Recibe las peticiones de generación de CAPTCHA		
Interfaces	Entrada: fichero XML con la petición de generación de CAPTCHA Salida: CAPTCHA (*java.awt.image.BufferedImage* API Java SE 6)		
Procesamiento	Recibe la petición de generación, envía la solicitud al sistema y espera hasta que el CAPTCHA se construye para devolverlo		
Datos	Procesa la petición, comprobando la firma del mensaje y desencriptandolo		
Manejo de Errores	Recoge los errores en la construcción del CAPTCHA e informa a la aplicación del usuario.		
Dependencias	Generador de CAPTCHAs		

Tabla 73. SecurityServerMain

Identificador	2.2.2 SSLServerConection	**Tipo**	Clase
Funcionalidad	Implementa la interfaz de comunicación con el componente del terminal.		
Interfaces	**Entrada:** entrada de datos conexión SSL. **Salida:** salida de datos conexión SSL.		

Procesamiento	Construye una socket seguro mediante el protocolo SSL con el servidor.
Datos	Todos los datos transmitidos a través de transmisión estarán provistos de integridad, confidencialidad y autenticación en el receptor.
Manejo de Errores	Informa al componente del terminal de los problemas de conexión con la el servidor.
Dependencias	Componente del terminal.

Tabla 74. SSLServerConection

Identificador	2.2.3 MessageParser	**Tipo**	Clase
Funcionalidad	Recoge la información del fichero XML y la traduce a objetos Java.		
Interfaces	Entrada: fichero XML. Salida: contenido de los elementos del XML.		
Procesamiento	Recorre un fichero XML, comprueba si se adecua al esquema acordad en el DTD y almacena los elementos válidos en objetos Java.		
Datos	-		
Manejo de Errores	Informa al servidor de los errores de construcción del fichero XML.		
Dependencias	Interfaz capa lógica.		

Tabla 75. MessageParser

5.2.2.1.3 Algoritmo

El proceso de recepción de peticiones de generación de CAPTCHAs induye la verificación de la petición por parte del receptor, el proceso seguido es el siguiente:

1. **Extracción de usuario.** Comprueba que conoce al usuario que ha hecho la petición, para ello accede al a base de datos y busca en la tabla de usuarios el id que se le ha proporcionada, si existe almacena su calve pública.
2. **Comprobación de firma.** Comprueba la firma del mensaje. Para ello el mismo verifica la firma ya que conoce la clave pública del usuario.
3. **Desencriptado mensaje.** Una vez comprobada la firma del mensaje, sabe que sólo una tarjeta perteneciente al sistema ha podido generar el mensaje, procede a descifralo con su clave privada y envía la petición al generador de Capuchas.

5.2.2.2 Subcomponente generador de CAPTCHAs

5.2.2.2.1 Arquitectura

Dado que el módulo deberá integrarse con el resto de la plataforma se ha diseñado la aplicación para que su integración y mantenimiento sea lo sencillo posible. Por estas razones se decidió realizar un desarrollo basado en una arquitectura de tres capas. Este patrón de diseño de tres niveles o capas está compuesto por:

- **La capa de presentación** que da al usuario (en este caso el servidor) acceso a la aplicación.

- **La capa lógica** controla las funcionalidades de la aplicación y dirige el flujo de datos. En esta capa los problemas críticos son solucionados y las funcionalidades de la aplicación se implementan construyendo una estructura robusta para el manejo de la base de datos.
- **La capa de datos** es el nivel que interactúa directamente con la base de datos. El nivel de datos gestiona el acceso a la base de datos y actúa de interfaz para el sistema.

Las características de la arquitectura en tres capas provee a nuestra aplicación de una serie de ventajas muy provechosas como es una alta independencia del resto de la plataforma, lo que facilitará las labores de integración y mantenimiento, desacopla el resto de módulos de la base de datos e incrementa la política de seguridad sobre el acceso a está sin que afecte a la percepción general del sistema de los datos y aportando mayor control sobre el manejo de datos, permite realizar futuras modificaciones en una de las capas sin afectar al resto de capas.

5.2.2.2.2 Descripción de componentes

Capa presentación

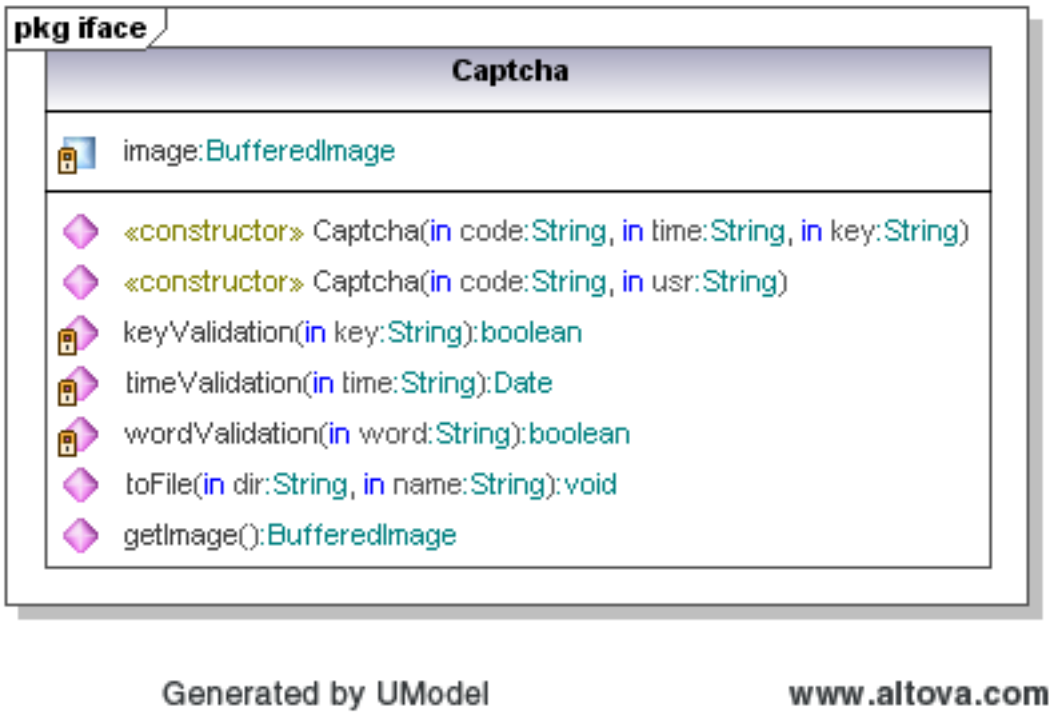

Figura 33. Diagrama de clases capa presentación

Identificador	2.2.1 iface	**Tipo**	Paquete
Funcionalidad	Capa vista. Comunica el sistema con el exterior.		
Interfaces	Entrada: petición generación del CAPTCHA. Salida: CAPTCHA (*java.awt.image.BufferedImage* API Java SE 6).		
Procesamiento	Recibe la petición de generación, envía la solicitud al sistema y espera hasta que el CAPTCHA se construye para devolverlo.		

Datos	Esta capa facilita el mantenimiento y seguridad del sistema. Implementación de patrón *Adapter* entre las peticiones del usuario y la capa lógica y se encarga de la validación de campos de entrada.
Manejo de Errores	Recoge los errores de la capa lógica y se los muestra al usuario del sistema.
Dependencias	Interfaz capa lógica.

Tabla 76.iface

Identificador	2.2.1.1 Captcha	**Tipo**	Clase
Funcionalidad	Implementación del patrón *Adapter*. Adapta la petición del usuario al formato de la capa lógica.		
Interfaces	Entrada: **Captcha(String mensaje, String usuario)** Salida: almacenamiento del CAPTCHA en la variable privada *image.* **getImage()** : devuelve un *java.awt.image.BufferedImage* **toFile(String dir, String nombre)** : crea un fichero "nombre.jpg" en el directorio *dir.*		
Procesamiento	Recibe los parámetros para la generación por cualquiera de las dos interfaces y se evalúan los parámetros de entrada. Envía la petición a la capa lógica y espera su respuesta.		
Datos	Se aplica a cada dato de entrada validaciones de formato para evitar ataques de tipo "command injection" que puedan acceder a la base de datos o al espacio de memoria propio del módulo. Cualquier problema de validación se devuelve un error. Una vez validados, se genera una petición de ticket que se enviará al a capa lógica.		
Manejo de Errores	Genera errores si existen problemas de validación y recoge los propios de las capas inferiores lanzándolos al sistema invocador.		
Dependencias	Interfaz capa lógica.		

Tabla 77. Captcha

Capa lógica

Identificador	2.2.2 Lógica	**Tipo**	Paquete
Funcionalidad	Capa lógica.		
Interfaces	Entrada: petición desde la capa de vista. Salida: CAPTCHA (*java.awt.image.BufferedImage* API Java SE 6).		
Procesamiento	Es donde se realizan las modificaciones en las imágenes para generar un CAPTCHA.		
Datos	Recibe la petición de la capa superior y realiza peticiones de imágenes a la capa de datos que luego modifica mediante el uso de librerías de filtros y de tratamiento de imágenes.		
Manejo de Errores	Recoge los errores generados en la capa lógica y los envía a la capa vista para que sean mostrados al usuario.		
Dependencias	Interfaz de la Capa de datos.		

Tabla 78. Capa lógica

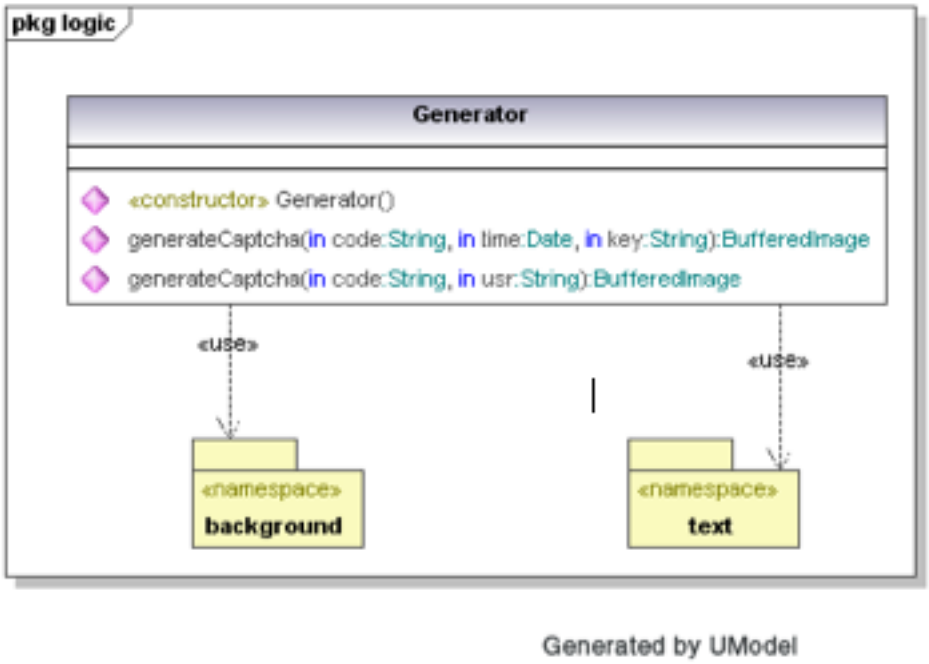

Figura 34. Diagrama de clases capa lógica

Identificador	2.2.2.1 Generador	**Tipo**	Clase
Funcionalidad	Genera CAPTCHAs a petición.		
Interfaces	Entrada: petición desde la capa de vista **generateCaptcha(String code, String usr)** Salida: CAPTCHA (*java.awt.image.BufferedImage* API Java SE 6)		
Procesamiento	Tutela la generación del CAPTCHA, realiza las llamadas a todos los involucrados en el proceso y coordina la secuencia del algoritmo de generación.		
Datos	Realiza la petición de imágenes a la capa de datos, se las facilita al generador de *background* y posteriormente llama al *textWriter* para que incluya en la imagen el mensaje y el sello de tiempo.		
Manejo de Errores	Recopila los errores producidos por las entidades a las que llama para generar el CAPTCHA.		
Dependencias	Interfaz de la Capa de datos, paquete org.visualCryptography.logic.background y paquete org.visualCryptography.logic.text		

Tabla 79. Generador

Identificador	2.2.2.2 Background	**Tipo**	Subpaquete
Funcionalidad	Generación de la imagen de fondo del CAPTCHA.		
Interfaces	Entrada: imágenes que compondrán el fondo del CAPTCHA. Salida: *background* del CAPTCHA.		

Procesamiento	Se compone el background mediante la aplicación de distintos tratamientos primero un formateo cada una de las imágenes por separado y posteriormente, una vez compuesto el mosaico, se realiza un tratamiento de distorsión sobre este.
Datos	Recibe las imágenes originales obtenidas de la base de datos se transforman hasta obtener el fondo de la imagen devolviendo el control del proceso a la clase Generator .
Manejo de Errores	Cualquier error será tratado internamente en el módulo y se devolverá una imagen nula si fuera oportuno.
Dependencias	API externo de tratamiento de imágenes jhLabs.

Tabla 80. Background

Identificador	2.2.2.2.1 BackgroundGenerador	**Tipo**	Clase
Funcionalidad	Tutela y guía la generación de la imagen de fondo.		
Interfaces	Entrada: imágenes (originales) que compondrán el fondo del CAPTCHA **Background(BufferedImage src_1, BufferedImage src_2)** Salida: almacenamiento de la imagen de fondo del CAPTCHA en la variable privada *image* (*java.awt.image.BufferedImage* API Java SE 6). **getImage() : devuelve un java.awt.image.BufferedImage**		
Procesamiento	Recibe las imágenes, comprueba que estas no son nulas, formatea cada una de ellas mediante las funcionalidades facilitadas por SubimageFormat, genera el mosaico y aplica dos transformaciones aleatorias de la clase DeformationList sobre la imagen final.		
Datos	-		
Manejo de Errores	Los errores provenientes de los distintos objetos invocados serán capturados y tratados, informando a los procesos superiores tan sólo si es necesario.		
Dependencias	Clase SubimageFormat, DeformationList y paquete Filters.		

Tabla 81. BackgroundGenerator

Identificador	2.2.2.2.2 SubimageFormat	**Tipo**	Clase
Funcionalidad	Aplica distintas transformaciones sobre las dimensiones de la imagen		
Interfaces	Entrada: imagen original (*java.awt.image.BufferedImage* API Java SE 6) y pixeles de altura que se desea que la imagen final tenga. **BufferedImage applyFormat(BufferedImage src, int height)** Salida: devuelve la nueva imagen con las dimensiones deseadas (*java.awt.image.BufferedImage* API Java SE 6).		
Procesamiento	Se aplica un corte entre un 1% a un 20% en uno de los lados de la imagen y se ajusta las dimensiones de esta a la altura adjunta en la petición mediante una función de zoom.		
Datos	La imagen original es modificada a lo largo de la secuencia de transformaciones, ninguna variable adicional es almacenada.		
Manejo de Errores	Cualquier error es lanzado a la clase BackgroundGenerator donde se tratará.		

Dependencias	Paquete org.visualCryptography.logic.background.filters.

Tabla 82. SubimageFormat

Identificador	2.2.2.2.3 DeformationList	**Tipo**	Clase
Funcionalidad	Contiene la lista de las deformaciones aplicadas a la imagen de *background*		
Interfaces	Entrada: imagen a transformar (*java.awt.image.BufferedImage*) y el número [0-9] de la transformación que se desea aplicar **BufferedImage applyDeformation(BufferedImage image, int id)** Salida: devuelve la nueva imagen (*java.awt.image.BufferedImage*)		
Procesamiento	Aplica la transformación seleccionada a la imagen de entrada y devuelve el resultado		
Datos	La imagen de entrada es pasada como parámetro al filtro. Los valores de modificación en cada uno de los filtros se escoge de forma aleatoria		
Manejo de Errores	Cualquier error es lanzado a la clase BackgroundGenerator donde se tratará		
Dependencias	Paquete org.visualCryptography.logic.background.filters.		

Tabla 83. DeformationList

Identificador	2.2.2.2.4 Filtros	**Tipo**	Librería
Funcionalidad	Contiene filtros propios.		
Interfaces	Entrada: imagen a transformar (*java.awt.image.BufferedImage*). Salida: devuelve la nueva imagen (*java.awt.image.BufferedImage*).		
Procesamiento	Compuesto por el API proporcionado por jhLabs iamge editor y otros de propia constructor.		
Datos	La imagen es modificada mediante la aplicación de funcionalidades derivadas de la librería **java.awt** (API Java SE 6).		
Manejo de Errores	Cualquier error es lanzado a la clase BackgroundGenerator donde se tratará.		
Dependencias	Paquete org.visualCryptography.logic.background.filters.		

Tabla 84. Filtros

Identificador	2.2.2.3 Texto	**Tipo**	Subpaquete
Funcionalidad	Contiene las funcionalidades para la escritura en imágenes.		
Interfaces	Entrada: petición de escritura sobre imagen. Salida: imagen con texto.		
Procesamiento	Introduce texto embebido en imágenes.		
Datos	Transforma el mensaje y el sello de tiempo de cadena de caracteres a imagen.		
Manejo de Errores	Los errores se reportan a la clase org.visualCryptography.logic.Generator donde se tratarán.		
Dependencias	Ninguna.		

Tabla 85. Filtros

Identificador	2.2.2.3.1 TextWriter	Tipo	Clase
Funcionalidad	Clase		
Interfaces	Escribe un texto sobre una imagen.		
Procesamiento	Entrada: petición de escritura en una imagen **BufferedImage addText(BufferedImage image, String word)** **BufferedImage addTimeStamp(BufferedImage image, String word)** Salida: imagen con texto (*java.awt.image.BufferedImage*)		
Datos	Se selecciona de forma aleatoria una fuente para el texto (invocación clase RandomFontGenerator), se comprueba que el tamaño del texto con la fuente seleccionada no sobrepase en ancho de la imagen, si esto sucede se procede a elegir otra fuente. Se elige de forma aleatoria la posición, teniendo en cuenta si existe texto insertado anteriormente y se adjunta a la imagen, esta se devuelve. Si lo que se quiere adjuntar es un sello de tiempo se deberá elegir un formato de forma aleatoria (clase Timestamp).		
Manejo de Errores	-		
Dependencias	Los errores se reportan a la clase org.visualCryptography.logic.Generator donde se tratarán.		

Tabla 86. TextWriter

Identificador	2.2.2.3.2 RandomFontGenerator	Tipo	Clase abstracta
Funcionalidad	Selecciona de forma aleatoria los valores de los parámetros que componen una fuente de letra (*java.awt.Font* API Java SE 6)		
Interfaces	Clase abstracta accedida desde sus hijos.		
Procesamiento	Selecciona de forma aleatoria la familia de la fuente, el tamaño y el estilo y comprueba que para la fuente seleccionada todos los caracteres son válidos (algunos caracteres especiales no existen en algunas fuentes).		
Datos	Se debe facilitar la palabra o letra sobre la que se aplicará la fuente para comprobar si existen los caracteres para esa fuente. Las familias de las fuentes se obtienen mediante. **java.awt.GraphicsEnvironmentet.AvailableFontFamilyNames();**		
Manejo de Errores	Los errores se reportan a la clase org.visualCryptography.logic.Generator donde se tratarán.		
Dependencias	Ninguna.		

Tabla 87. RandomFontGenerator

Identificador	2.2.2.3.3 TextFont	Tipo	Clase
Funcionalidad	Clase		
Interfaces	Selecciona de forma aleatoria los valores de los parámetros que componen una fuente de letra (*java.awt.Font* API Java SE 6).		
Procesamiento	Entrada: petición de nueva fuente, proporcionando la palabra de aplicación para poder comprobar si la fuente es válida para sus caracteres		

	Font nextFont(String word) Salida: tipo de fuente (*java.awt.Font* API Java SE 6).
Datos	Hereda de org.visualCryptography.logic.text.RandomFontGenerator. Aporta funcionalidades de deformación de la fuente como escritura en ángulo, pessimaPrint y ScatterType (técnicas aplicadas en la generaciónd e CAPTCHAs basados en texto).
Manejo de Errores	Se establece el tamaño de la fuente como máximo de 50 y como mínimo de 30.
Dependencias	Los errores se reportan a la clase org.visualCryptography.logic.Generator donde se tratarán.

Tabla 88. TextFont

Identificador	2.2.2.3.4 TimestampFont	**Tipo**	Clase
Funcionalidad	Selecciona de forma aleatoria los valores de los parámetros que componen una fuente de letra (*java.awt.Font* API Java SE 6).		
Interfaces	Entrada: petición de nueva fuente, proporcionando la palabra de aplicación para poder comprobar si la fuente es válida para sus caracteres **Font nextFont(String word)** Salida: tipo de fuente (*java.awt.Font* API Java SE 6).		
Procesamiento	Hereda de org.visualCryptography.logic.text.RandomFontGenerator. Reescribe el método padre de selección de estilo obligando a que este sea en negrita (con o sin cursiva).		
Datos	Permite estableces mediante el constructor el tamaño mínimo y máximo de fuente		
Manejo de Errores	Los errores se reportan a la clase org.visualCryptography.logic.Generator donde se tratarán.		
Dependencias	Ninguna		

Tabla 89. TimestampFont

Identificador	2.2.2.3.4 Timestamp	**Tipo**	Clase
Funcionalidad	Selecciona de forma aleatoria un formato para el sello de tiempo		
Interfaces	Entrada: tres de interfaces de petición. Sin parámetros. Toma el tiempo del sistema mediante un objeto Calendar (*java.util.Calendar* API Java SE 6) **String applyRandomDateFormat()** Petición con el sello de tiempo. Recibe un objeto de tipo Date (*java.util.Date* API Java SE 6) **String applyRandomDateFormat(Date timeStamp)** Petición con el sello de tiempo y formato especifico. Recibe un objeto de tipo Date (*java.util.Date* API Java SE 6) y el formato en el que se quiere mostrar **String applyRandomDateFormat(String pattern, Date timeStamp)** Salida: devuelve un String con el sello de tiempo con un nuevo formato.		
Procesamiento	Selecciona uno de los 160 formatos de tiempo disponible y lo aplica a un		

	objeto Date transformándolo a String.
Datos	Todos los formatos de sello de tiempo se contienen día, mes, año, hora, minuto y segundo.
Manejo de Errores	Los errores se reportan a la clase org.visualCryptography.logic.Generator donde se tratarán.
Dependencias	Ninguna.

Tabla 90. Timestamp

Capa de datos

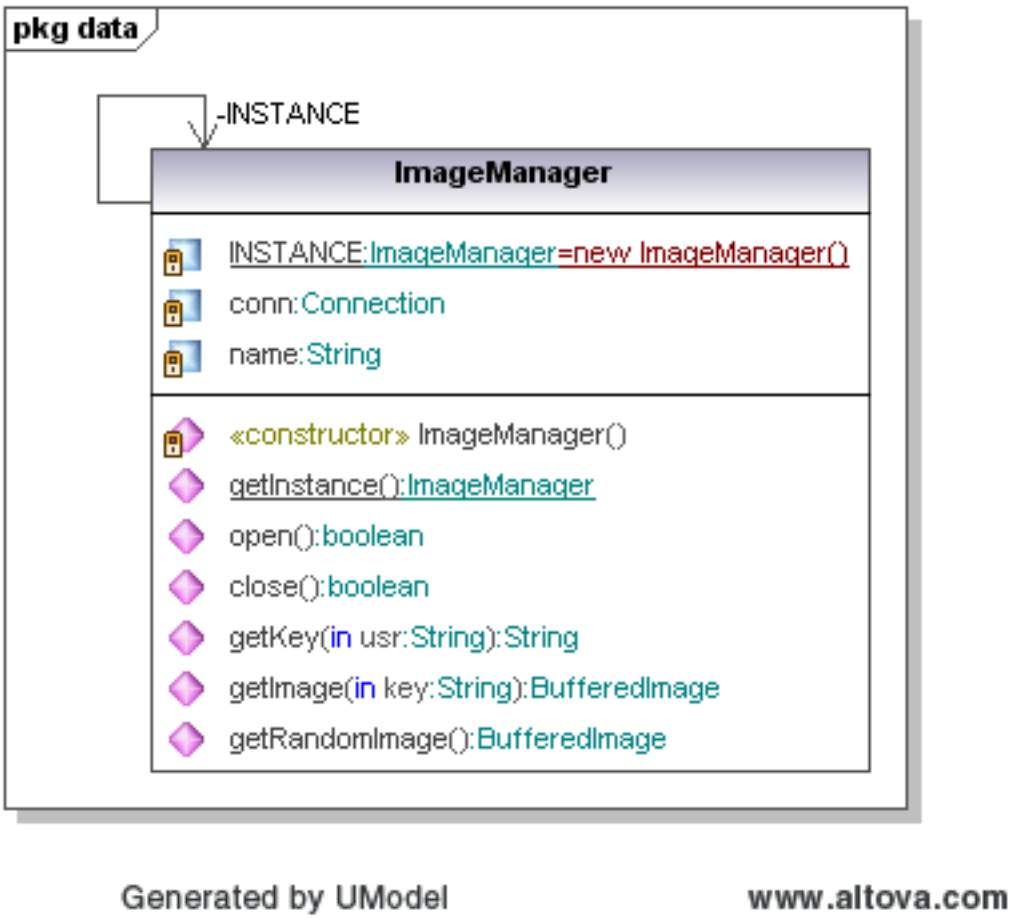

Figura 35. Diagrama de calses de la capa de datos

Identificador	2.2.3 Datos	**Tipo**	Paquete
Funcionalidad	Acceso a la base de datos.		
Interfaces	Entrada: realiza una petición mediante el objeto estático. **ImageManager.getInstance()** Salida: respuesta de la base de datos.		
Procesamiento	En esta capa se establece una conexión única a la base de datos mediante el driver JDBCPostgresDriver. A través de esta conexión se realizan las consultas sobre la base de datos. El sistema sólo debe tener permisos de		

	consulta.
Datos	-
Manejo de Errores	Recoge los errores de acceso a la base de datos y los trasmite a las capas superiores.
Dependencias	Base de datos.

Tabla 91. Capa de datos

Identificador	2.2.3.1 ImageManager	**Tipo**	Clase
Funcionalidad	Accede a la base de datos.		
Interfaces	Entrada: petición de imagen. Existen dos tipos de peticiones. Petición de imagen etiquetada en por un concepto **BufferedImage getImage(String key)** Petición de una imagen de una imagen aleatoria de la base de datos **BufferedImage getRandomImage()** Salida: imagen (*java.awt.image.BufferedImage*)		
Procesamiento	Se realizan comandos postgresql para la selección de imágenes dentro de la base de datos.		
Datos	La base de datos devuelve un array de bytes que representa la imagen y que se convierte a *java.awt.image.BufferedImage* formato que maneja el sistema.		
Manejo de Errores	Recoge los errores de acceso a la base de datos y los trasmite a las capas superiores.		
Dependencias	Base de datos.		

Tabla 92. ImageManager

5.2.2.2.3 Algoritmo

En esta sección se pretende explicar de forma muy esquemática el funcionamiento y pasos que sigue el algoritmo que hemos diseñado para la generación de CAPTCHAs autenticados mediante criptografía visual.

1. **Selección de imágenes**
   1.1. Selección de forma aleatoria de una imagen de la base de datos cuya etiqueta coincida con la clave entre los comunicantes
   1.2. Selección de una imagen aleatoria en la base de datos
2. **Composición del mosaico**
   2.1. Formateado de las imágenes por separado
      2.1.1. Corte de uno de sus lados (elegido de forma aleatoria) de 1- 20% de su tamaño. Aunque la aplicación de esta transformación, si una máquina posee la imagen original, no disminuye la probabilidad de reconocimiento de la imagen, su aplicación añade aleatoriedad al mensaje final, evitando que se generen dos mensajes exactos.
      2.1.2. Rescaldado de la imagen para ajustar su dimensión a 125 pixeles de altura. Ambas imágenes deben concordar en altura sea cual sea su tamaño original para la armonía de

su unión. Se ha elegido 125 por la equiparación entre tamaño en disco (no más de 15K) y dimensión de visualización aunque esta última siempre dependerá de las características de la pantalla del usuario. Este parámetro se puede cambiar en cualquier momento.

2.2. Composición del mosaico

2.2.1. Unión de ambas imágenes en orden aleatorio.

2.2.2. Aplicación de filtros de ecualización de brillo para que sea más complejo de identificar la división entre las imágenes.

Figura 36. Esquema de generación de CAPTCHAs

3. Aplicación de transformaciones al background

3.1. Aplicación de filtro para aumentar luminancia (GammaFilter, Jhlabs API). Aumenta los blancos de la imagen y permite que las letras del mensaje contrasten con el fondo

3.2. Aplicación de dos filtro elegidos de forma aleatoria de la siguiente lista

- Difusión (DiffuseFilter, Jhlabs API). La imagen se difumina al difundir cada pixel en una dirección aleatoria. La longitud del movimiento se pasa como parámetro.
- Difuminación (DiffusionFilter, Jhlabs API). Este filtro utiliza el método del error de distorsión de Floyd-Steinberg para difuminar el tono de los colores de la imagen
- Desplazamiento (DisplaceFilter, Jhlabs API). Cada uno de los pixeles de la imagen original se desplaza en la escala de grises de acuerdo con el mapa de desplazamiento que se le pasa, como resultado la imagen simula ser vista a través de un cristal.

- Distorsión (DitherFilter, Jhlabs API). Filtro que distorsiona de forma simétrica toda la imagen.
- Desenfoque Gaussiano (GaussianFilter, Jhlabs API). Filtro que crea un desenfoque basándose en una distribución de Gauss.
- Efecto mármol (MarbleFilter, Jhlabs API). Filtro que trata de simular en efecto del mármol en la imagen destacando zonas de la imagen como si fueran protuberancias.
- Cuantización del color (QuantizerFilter, Jhlabs API). El proceso de cuantización tiene como objeto reducir el número de colores diferentes que se emplean en una imagen.
- Posterización (PosterizeFilter, Jhlabs API) La posterización de una imagen es la reducción de su gama tonal a unos pocos colores planos, uniformes.
- Efecto oleo (SmearFilter, Jhlabs API). Este filtro dispersa los pixeles a sus zonas adyacentes consiguiendo distintos efectos pintura al óleo. Se puede elegir forma de cruces, líneas, cuadrados y círculos y cambiar los parámetros como el tamaño de las formas y el ángulo.

4. Escritura del mensaje y sello de tiempo

4.1. Inclusión del mensaje

4.1.1. Selección de una fuente de letra aleatoria para cada una de los caracteres. La fuente incluye tipo letra, tamaño y color[4]. Verifica si el texto cabe dentro de la imagen, si no vuelve a seleccionar fuente. Más de 1000 fuentes distintas para elegir.

4.1.2. Escritura del mensaje. Selección aleatoria de la posición en la imagen

4.2. Inclusión del sello de tiempo

4.2.1. Selección de forma aleatoria del formato de fecha y horas. Más de 100 variaciones.

4.2.2. Selección de una fuente de forma aleatoria para todo el timestamp. La fuente incluye tipo letra, tamaño y color[4].

4.2.3. Inclusión del sello en el espacio dejado por el mensaje, dependiendo de la localización de este.

5.2.2.3 Subcomponente base de datos

La base de datos es uno de los componentes más relevantes del servidor de generación de CAPTCHAs. Una gestión eficiente y segura de la misma ayudará en gran medida al éxito de la generación de mensajes seguros.

5.2.2.3.1 Arquitectura

PostgreSQL es un potente sistema de gestión de bases de datos relacionales orientadas a objetos de código abierto. Una de sus principales ventajas es que es multiplataforma, puede ser utilizado en los principales sistemas operativos Linux, UNIX AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64) y Windows. Es totalmente compatible con ACID, tiene soporte completo para claves foráneas, uniones, vistas, disparadores y procedimientos almacenados (en varios idiomas). Se incluye la mayor parte de SQL: 2008 tipos de datos, incluyendo INTEGER, NUMERIC, Boolean, CHAR, VARCHAR, DATE, INTERVAL, y

[4] Aunque para el ojo humano el color de las letras parece que es negro, el algoritmo elige de forma aleatoria un color en la gama de los grises oscuros lo que añade aleatoriedad a la imagen y hace que una máquina tenga más dificultades en reconocer las zonas de texto.

TIMESTAMP. También soporta almacenamiento de objetos binarios grandes, como imágenes, sonidos o vídeo. Cuenta con interfaz de programación de aplicaciones Java (PostgreSQL JDBC Driver) y muchos otros lenguajes como C / C + +, .Net, Perl, Python y Ruby entre otros. Es altamente escalable, tanto en la gran cantidad de datos que puede manejar y en el número de usuarios concurrentes que puede acomodar. Permite almacenar hasta 32TB de datos en cada tabla y 1 GB en cada campo.

Mediante un modelo entidad relación, Figura 37, se ha querido representar la información que la base de datos deberá contener. Básicamente necesitamos una estructura que almacene los objetos de tipo imagen y que asocie a ellos un número variable de etiquetas que representarán calves que cada uno de los destinarios de los mensajes han acordado con el sistema. Sería interesante, y recomendable, integrar nuestra base de datos en la que utiliza el sistema, lo que tan sólo implicaría que se añadiera un nuevo campo a la tupla de cada usuario que referenciara a la clave acordada.

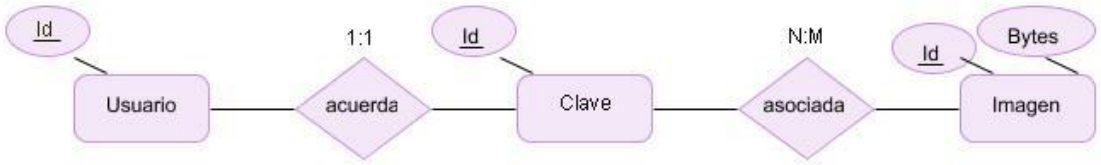

Figura 37. Modelo entidad-relación

El punto más destacable de la base de datos son las imágenes que representan los conceptos asociados como claves. Dado que la autenticación del mensaje se basará en la percepción que el usuario tenga de la imagen, estas deben cumplir una serie de requisitos:

- El concepto u objeto debe situarse en primer plano de la imagen y a poder ser que sea el único objeto reconocible de la misma.
- Evitar imágenes artísticas, es decir, aquellas que sólo contienen una parte del objeto, que están tomadas desde ángulos raros, en las que el perfil del objeto no es claro o reconocible o aquellas que tienen algún tipo de efecto de luces, reflejos, etc.
- El fondo de la imagen debe de ser de un color distinto al del objeto, evitando también fondos negros o imágenes en blanco y negro.
- La imagen debe ser rectangular siendo más ancha que larga, se formaliza esta restricción haciendo que todos las imágenes cumplan que su ancho deba ser igual o mayor que 1’25 veces su altura.

Para cada imagen se deberá tener un identificador o nombre de la imagen, la propia imagen se almacena una colección de bytes, junto con las etiquetas que describen la imagen. Una imagen puede tener asociada más de una etiqueta, lo propio es almacenar en una tabla aparte las relaciones entre las claves-etiquetas y en otra las imágenes. Con la estructura de dos tablas descrita se obtendría una base de datos simple pero que cumple las especificaciones del problema, sin embargo, con el conocimiento que tenemos sobre el acceso a la tablas aconsejamos añadir una tercera tabla que recoja todas las claves y el número de imágenes que hay etiquetadas con esa clave. Esta tabla colaborará en la selección de una imagen etiquetada con una clave concreta haciendo la operación más rápida. Aunque

postgreSQL reconoce la operación COUNT sobre una tabla para realizar el recuento de todas la entradas, esta funcionalidad es muy lenta, lo que agravaría en gran medida el tiempo de ejecución del módulo.

En la Figura 38 se muestra gráficamente, mediante un modelo relacional el diseño de la base de datos.

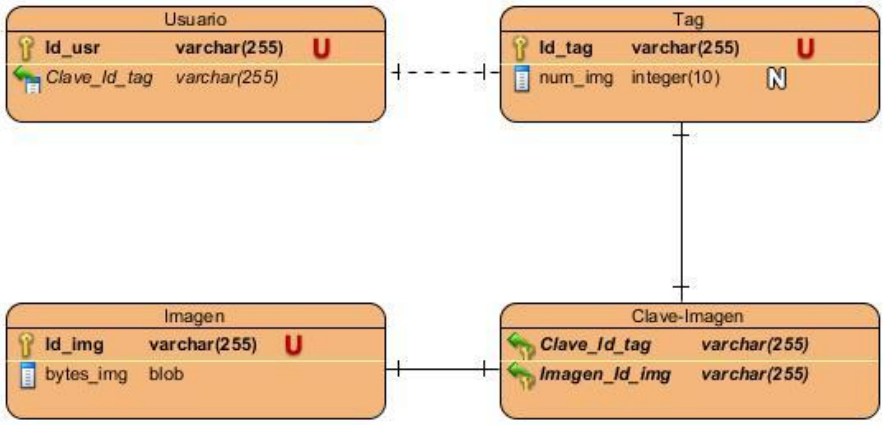

Figura 38. Modelo relacional

5.2.2.3.2 Componentes

A continuación se describe de forma más explícita el contenido de cada una de las tablas que se deberán de construir en la base de datos:

Tabla Imagen		
Id_img	text (no limitado, tipo propio de postgres)	Clave primaria
Bytes_img	bytea (tipo datos binarios)	Imagen

Tabla 93. Tabla Imagen de la base de datos

Tabla Clave-Imagen		
Id_clave	text (no limitado, tipo propio de postgres)	Clave primaria
Id_imagen	text (no limitado, tipo propio de postgres)	

Tabla 94. Tabla Clave-imagen de la bae de datos

Tabla Clave (conceptos)		
Id_clave	text (no limitado, tipo propio de postgres)	Clave primaria
Num_img	Integuer (dato numérico de 4 bytes)	Número de imagen etiquetadas por *id_clave*

Tabla 95. Tabla Clave de la base de datos

Tabla Usuario (conceptos)		
Id_usr	text (no limitado, tipo propio de postgres)	Clave primaria
Id_clave	text (no limitado, tipo propio de postgres)	Clave del usuario

Tabla 96. Tabal Usuario de la base de datos

5.2.2.3.3 Algoritmo

No aplica.

5.2.3 Componente de tarjeta inteligente

5.2.3.1 Arquitectura

La arquitectura básica de una JavaCard consiste de: applets, JavaCard API, maquina virtual de Javacard, JCRE y el sistema operativo de la tarjeta.

- **Applets.** Los applets son las aplicaciones que corren embarcadas en una JavaCard.
- **JavaCard API.** Contiene la definición de las clases requeridas para utilizar la JCVM y el JCRE.
- **JCVM**. Necesario para la ejecución del bytecode de javacard**.**
- **JCRE.** Comprende API, JCVM, sobre el se ejecutan los applets desarrollados.
- **COS:** Sistema operativo nativo de la tarjeta.

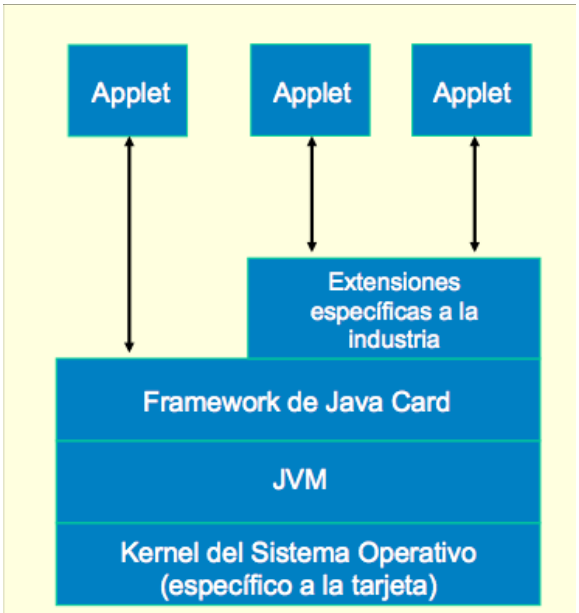

Figura 39. Arquitectura JavaCard

La implementación desarrollada para el sistema consiste en un único applet ya que sólo se requiere su uso para una persona. La aplicación consiste en una pequeña clase que contiene los datos necesarios para realizar las operaciones criptográficas demandadas a la tarjeta: Identificador usuario, par de claves del usuario para poder firmar en su nombre y clave pública del servidor para poderle enviar mensajes

encriptados; y realiza operaciones de verificación de pin, generación de número aleatorio de 4 dígitos (OTP), firma de mensaje, encriptado de mensajes mediante clave pública del servidor y verificación de OTP.

5.2.3.2 Componentes

No aplica.

5.2.3.3 Algoritmo

No aplica.

Capítulo 6

PLAN DE PRUEBAS

En este capítulo se definirá el plan de pruebas que se han realizado en el proyecto. El objetivo perseguido por el plan de pruebas es el de comprobar que las necesidades de los usuarios del sistema han sido satisfechas. Para ello, se describirán las tareas necesarias para las pruebas, el entorno de pruebas, así como los criterios de aceptación de las mismas, finalmente se mostrará una tabla resumen con los resultandos obtenidos en el plan de pruebas.

6.1 Elementos de prueba

Con el fin de comprobar que el sistema satisface las necesidades de los usuarios, los elementos sujetos a prueba son los requisitos de usuario de capacidad definidos en el capítulo 2 Requisitos de Usuario, los cuales detallan las funcionalidades que el sistema debe proporcionar. Concretamente los elementos susceptibles de pruebas serán:

- **RUC-001:** Autenticación de usuario.
- **RUC-002:** Firmar mensaje.
- **RUC-003:** Verificación de mensaje.
- **RUC-004:** Finalizar operación.
- **RUC-005:** Fin automático de la operación.

6.2 Características que se probarán

En el siguiente apartado se identificarán las características que se probarán para cada uno de los elementos listados en la sección anterior. El criterio que se ha seguido para clasificar las características ha sido el siguiente:

- **Funcionalidad:** Mediante el requisito se puede probar alguna de las funcionalidades descritas en un requisito de usuario de capacidad.
- **Sistema:** La prueba del requisito afecta a dos o varios requisitos de usuario simultáneamente.
- **Datos:** La prueba del requisito comprueba el mantenimiento y generación de los datos de la aplicación.

En la tabla que se muestra a continuación, se describen las características que se probarán de cada uno de los elementos:

Elemento de Prueba	Descripción del elemento	Característica a probar
RUC-001	Autenticar unívocamente al propietario/usuario de la tarjeta inteligente en el proceso de login.	Sistema
RUC-002	Realizar la firma digital sobre un mensaje introducido por el usuario.	Datos
RUC-003	Verificar unívocamente el mensaje antes de firmarlo.	Funcionalidad
RUC-004	Finalizar operación a petición del usuario	Funcionalidad
RUC-005	Finalizar automáticamente la operación al desconectar la tarjeta.	Sistema

Tabla 97. Elementos de pruebas

6.3 Criterio de aceptación/rechazo de una prueba

El ámbito de aceptación general para validar una determinada prueba, será la de satisfacer los requisitos de usuario que están relacionados con dicha prueba. Para ello, en el siguiente apartado se detallarán cada una de las pruebas a realizar, así como los criterios de aceptación concretos.

6.4 Descripción de las pruebas

A continuación se especificarán los casos de prueba posibles para llevar a cabo el plan de pruebas. En estos casos de prueba se detallará:

1. Un **identificador** único para cada caso de prueba, siendo de la forma CP-XX y representado XX dos dígitos que comenzarán en el 01 y crecerán de manera secuencial por cada caso de prueba.
2. Los **elementos de prueba** asociados a la prueba y que han sido recogidos en el punto "6.1 Elementos de prueba".
3. **Propósito**, objetivo que se pretende cumplir con la prueba.
4. Las **especificaciones de entrada** para el caso de prueba, es decir que elementos software o acciones se necesitan previamente para realizar la prueba correctamente.
5. Las **especificaciones de salida**, o lo que es lo mismo, la salida esperada para cada caso de prueba.
6. Necesidades del **entorno** para realizar la prueba concreta.
7. **Acciones a realizar** para llevar a cabo la prueba.
8. **Criterio de aceptación/rechazo especifico** para cada prueba.

6.4.1 Relación de Pruebas

Identificador	*CP-01*	**Elemento en prueba**	RUC-001, RUR-002, RUR-005
Propósito	Verificar que la identificación de usuarios es unívoca.		
Especificaciones de entrada	• La tarjeta inteligente del usuario fue configurada: se almacenó el identificador del usuario, la clave privada del sevidor, y las claves del usuario. • La tarjeta inteligente de un segundo usuario fue configurada también. • Se almacenarón los identificadores y las claves públicas de ambos usuarios en el servidor del sistema.		
Especificaciones de salida	Los usuarios tienen acceso al sistema.		
Necesidades del entorno	Se requerirá un equipo con el sistema instalado y ejecutando. Este equipo tendrá conectado un lector de tarjetas inteligentes. Así mismo, se necesitará una tarjeta inteligente JavaCard.		
Acciones a realizar	• Arrancar el equipo con el lector de tarjetas conectado. • Introducir la tarjeta inteligente con el certificado del usuario. • Introducir el código PIN correspondiente a dicha tarjeta inteligente. • Comprobar que el sistema reconocer el PIN introducido. • Reiniciar el equipo con el lector de tarjetas conectado. • Introducir la tarjeta inteligente del segundo usuario. • Introducir el código PIN correspondiente a dicha tarjeta inteligente. • Comprobar que se le permite introducir un mensaje con esta segunda cuenta.		
Criterios de aceptación	La prueba será satisfactoria si cada tarjeta permite hacer el proceso de iniciar proceso de firma.		

Tabla 98. Caso de prueba 01

Identificador	*CP-02*	**Elemento en prueba**	RUC-001, RUR-002
Propósito	Comprobar que no es posible un proceso de autenticación exitosa sin introducir el número PIN de la tarjeta.		
Especificaciones de entrada	• La tarjeta inteligente del usuario fue configurada: se almacenó el identificador del usuario, la clave privada del sevidor, y las claves del usuario. • Se almacenó el identificadore del usuario y las calves públicas de ambos usuarios en el servidor del sistema.		
Especificaciones de salida	El usuario no tiene acceso al sistema.		
Necesidades del entorno	Se requerirá un equipo con el sistema instalado y ejecutando. Este equipo tendrá conectado un lector de tarjetas inteligentes. Así mismo, se necesitará una tarjeta inteligente JavaCard.		
Acciones a realizar	• Arrancar el equipo con el lector de tarjetas conectado. • Introducir la tarjeta inteligente. • Intentar acceder al sistema.		
Criterios de aceptación	La prueba será satisfactoria si el proceso de autenticación finaliza de forma fallida.		

Tabla 99. Caso de prueba 02

Identificador	*CP-03*	**Elemento en prueba**	RUC-001, RUR-006
Propósito	Comprobar que no es posible un proceso de autenticación exitosa sin conocer el número PIN de la tarjeta.		
Especificaciones de entrada	• La tarjeta inteligente del usuario fue configurada: se almacenó el identificador del usuario, la clave privada del sevidor, y las claves del usuario. • Se almacenó el identificadore del usuario y las calves públicas de ambos usuarios en el servidor del sistema.		
Especificaciones de salida	El usuario no tiene acceso al sistema.		
Necesidades del entorno	Se requerirá un equipo con el sistema instalado y ejecutando. Este equipo tendrá conectado un lector de tarjetas inteligentes. Así mismo, se necesitará una tarjeta inteligente JavaCard.		
Acciones a realizar	• Arrancar el equipo con el lector de tarjetas conectado. • Introducir la tarjeta inteligente. • Introducir un número PIN aleatorio. • Intentar acceder al sistema.		
Criterios de aceptación	La prueba será satisfactoria si el proceso de autenticación finaliza de forma fallida.		

Tabla 100. Caso de Prueba 03

Identificador	*CP-04*	**Elemento en prueba**	RUR-004
Propósito	Comprobar que la tarjeta y el sistema se bloquean al introducir 3 números PIN incorrectos consecutivamente.		
Especificaciones de entrada	• La tarjeta inteligente del usuario fue configurada: se almacenó el identificador del usuario, la clave privada del sevidor, y las claves del usuario. • Se almacenó el identificadore del usuario y las calves públicas de ambos usuarios en el servidor del sistema.		
Especificaciones de salida	El usuario no tiene acceso al sistema y la tarjeta inteligente se ha bloqueado.		
Necesidades del entorno	Se requerirá un equipo con el sistema instalado y ejecutando. Este equipo tendrá conectado un lector de tarjetas inteligentes. Así mismo, se necesitará una tarjeta inteligente JavaCard.		
Acciones a realizar	• Arrancar el equipo con el lector de tarjetas conectado. • Introducir la tarjeta inteligente. • Introducir un número PIN aleatorio. • Intentar acceder al sistema. • Sacar la tarjeta inteligente. • Introducir la tarjeta inteligente. • Introducir un número PIN aleatorio. • Intentar acceder al sistema. • Sacar la tarjeta inteligente. • Introducir la tarjeta inteligente. • Introducir un número PIN aleatorio. • Intentar acceder al sistema. • Sacar la tarjeta inteligente. • Introducir la tarjeta inteligente.		
Criterios de aceptación	La prueba será satisfactoria si el proceso de autenticación finaliza de forma fallida y el sistema no permite volver a introducir un número PIN.		

Tabla 101. Caso de prueba 04

Identificador	*CP-05*	**Elemento en prueba**	RUC-002, RUC-003, RUR-001, RUR-001, RUR-007, RUR-008, RUR-009
Propósito	Comprobar que es posible firmar un mensaje.		
Especificaciones de entrada	• La tarjeta inteligente del usuario fue configurada: se almacenó el identificador del usuario, la clave privada del sevidor, y las claves del usuario. • Se almacenó el identificadore del usuario y las calves públicas de ambos usuarios en el servidor del sistema.		
Especificaciones de salida	Se generá un fichero con el mensaje y su firma digital.		
Necesidades del entorno	Se requerirá un equipo con el sistema instalado y ejecutando. Este equipo tendrá conectado un lector de tarjetas inteligentes. Así mismo, se necesitará una tarjeta inteligente JavaCard.		
Acciones a realizar	• Arrancar el equipo con el lector de tarjetas conectado.		

	• Introducir la tarjeta inteligente. • Introducir número PIN. • Introducir mensaje. • Comprobar que el texto que aparece en el CAPTCHA es el mismo que se ha introducido. • Comprobar que una de las imágenes que aparecen en el CAPTCHA tiene relación con la clave establecida con el sistema. • Introducir código de verificación que aparece en el CAPTCHA.
Criterios de aceptación	La prueba será satisfactoria si se puede verifica la firma que aparece en el fichero como una firma del mensaje introducido y perteneciente al usuario de la tarjeta.

Tabla 102. Caso de prueba 05

Identificador	*CP-06*	**Elemento en prueba**	RUC-003, RUR-008
Propósito	Comprobar que no es posible firmar un mensaje si no se introduce en código de confirmación correcto.		
Especificaciones de entrada	• La tarjeta inteligente del usuario fue configurada: se almacenó el identificador del usuario, la clave privada del sevidor, y las claves del usuario. • Se almacenó el identificadore del usuario y las calves públicas de ambos usuarios en el servidor del sistema.		
Especificaciones de salida	El usuario no obtiene la firma del mensaje.		
Necesidades del entorno	Se requerirá un equipo con el sistema instalado y ejecutando. Este equipo tendrá conectado un lector de tarjetas inteligentes. Así mismo, se necesitará una tarjeta inteligente JavaCard.		
Acciones a realizar	• Arrancar el equipo con el lector de tarjetas conectado. • Introducir la tarjeta inteligente. • Introducir número PIN. • Introducir mensaje. • Comprobar que el texto que aparece en el CAPTCHA es el mismo que se ha introducido. • Comprobar que una de las imágenes que aparecen en el CAPTCHA tiene relación con la clave establecida con el sistema. • Introducir un número aleatorio como código de verificación.		
Criterios de aceptación	La prueba será satisfactoria si no se genera ningún fichero con la firma y se cierra la aplicación.		

Tabla 103. Caso de prueba 06

Identificador	*CP-07*	**Elemento en prueba**	RUC-003, RUR-007
Propósito	Comprobar que no el usuario es capaz de reconocer que un CAPTCHA no está generado por su tarjeta inteligente.		
Especificaciones de entrada	• La tarjeta inteligente del usuario fue configurada: se almacenó el identificador del usuario, la clave privada del sevidor, y las claves del usuario. • La tarjeta inteligente de un segundo usuario fue configurada		

	también. • Se almacenarón los identificadores y las claves públicas de ambos usuarios en el servidor del sistema.
Especificaciones de salida	El usuario no obtiene la firma del mensaje.
Necesidades del entorno	Se requerirá dos equipos con el sistema instalado y ejecutando. Estos equipos tendrán conectado un lector de tarjetas inteligentes. Así mismo, se necesitarán dos tarjetas inteligentes JavaCard.
Acciones a realizar	• Arrancar un equipo con el lector de tarjetas conectado. • Introducir una tarjeta inteligente. • Introducir número PIN. • Introducir mensaje. • Hacer dormir al sistema. • Arrancar el otro equipo equipo con el lector de tarjetas conectado. • Introducir la otra tarjeta inteligente. • Introducir número PIN. • Introducir el mismo mensaje. • Enviar el CAPTCHA obtenido al primer equipo • Comprobar que el texto que aparece en el CAPTCHA es el mismo que se ha introducido • Comprobar que una de las imágenes que aparecen en el CAPTCHA tiene relación con la clave establecida con el sistema.
Criterios de aceptación	La prueba será satisfactoria si el usuario no introduce el código de confirmación.

Tabla 104. Caso de prueba 07

Identificador	*CP-08*	**Elemento en prueba**	RUC-004
Propósito	Comprobar que el proceso de firma se fianliza cuando el usuario desea salir de la sesión.		
Especificaciones de entrada	• La tarjeta inteligente del usuario fue configurada: se almacenó el identificador del usuario, la clave privada del sevidor, y las claves del usuario. • Se almacenó el identificadore del usuario y las calves públicas de ambos usuarios en el servidor del sistema.		
Especificaciones de salida	El usuario sale del sistema.		
Necesidades del entorno	Se requerirá un equipo con el sistema instalado y ejecutando. Este equipo tendrá conectado un lector de tarjetas inteligentes. Así mismo, se necesitará una tarjeta inteligente JavaCard.		
Acciones a realizar	• Arrancar un equipo con el lector de tarjetas conectado. • Introducir una tarjeta inteligente. • Introducir número PIN. • Introducir mensaje. • Comprobar que el texto que aparece en el CAPTCHA es el mismo que se ha introducido • Comprobar que una de las imágenes que aparecen en el CAPTCHA tiene relación con la clave establecida con el sistema.		

	• Introducir secuencia de cierre de sesión.
Criterios de aceptación	La prueba será satisfactoria si el usuario es capaz de salir del sistema.

Tabla 105. Caso de prueba 08

Identificador	*CP-09*	**Elemento en prueba**	RUC-005
Propósito	Comprobar que el proceso de firma se fianliza cuando la tarjeta inteligente se saca del lector.		
Especificaciones de entrada	• La tarjeta inteligente del usuario fue configurada: se almacenó el identificador del usuario, la clave privada del sevidor, y las claves del usuario. • Se almacenó el identificadore del usuario y las calves públicas de ambos usuarios en el servidor del sistema.		
Especificaciones de salida	El usuario sale del sistema.		
Necesidades del entorno	Se requerirá un equipo con el sistema instalado y ejecutando. Este equipo tendrá conectado un lector de tarjetas inteligentes. Así mismo, se necesitará una tarjeta inteligente JavaCard.		
Acciones a realizar	• Arrancar un equipo con el lector de tarjetas conectado. • Introducir una tarjeta inteligente. • Introducir número PIN. • Introducir mensaje. • Sacar tarjeta inteligente.		
Criterios de aceptación	La prueba será satisfactoria si el usuario se ha salido del sistema.		

Tabla 106. Caso de prueba 09

6.5 Tabla resumen de resultados

	Satisfactorio	No satisfactorio	Inconcluyente
CP-01	X		
CP-02	X		
CP-03	X		
CP-04	X		

	Satisfactorio	No satisfactorio	Inconcluyente
CP-05	X		
CP-06	X		
CP-07	X		
CP-08	X		
CP-09	X		

Tabla 107. Resumen resultados pruebas

Capítulo 7

CONCLUSIONES Y FUTUROS DESARROLLOS

7.1 Conclusiones

En este capítulo se pretende describir las conclusiones extraídas durante la elaboración de este proyecto, así como, las líneas de posibles trabajos futuros que podrían derivarse a partir de éste.

7.1.1 Conclusiones generales

Durante este trabajo se destaca el papel de las tarjetas inteligentes en la seguridad de la información en el mundo global actual, dadas las funcionalidades avanzadas para la implementación de soluciones a través de los algoritmos criptográficos disponibles habitualmente en las librerías de los sistemas operativos de las smart cards. Al mismo tiempo, el proceso de aproximación a esta tecnología llevada a cabo para este proyecto nos muestra las capacidades hardware de este tipo de dispositivos que en la actualidad incluyen un conjunto de módulos dedicados exclusivamente a funciones de seguridad y que les convierten en elementos muy importantes dentro de numerosos sistemas.

Una de las aplicaciones más relevantes de esta técnología es que permite a cualquier persona sin extensos conocimientos criptográficos hacer uso de herramientas de seguridad muy

pontentes y de una forma sencilla. Por lo tanto se abre un nuevo campo en las técnologas de la información que hasta ahora no había llegado al usuario común y que permitirá incrementar, en particular, la seguridad en las comunicaciones diarias entre las personas a través de la red y, a nivel general, la seguridad de los sistemas informáticos donde, cada día, se protegen más los datos y procesos implicados en ellos.

Remarcar la aportación del protocolo desarrollado a lo largo de este proyecto dentro del uso de las tarjetas inteligentes. La solución ideada no solo permitirá al usuario disfrutar de una forma más gráfica de las funcionalidades criptográficas que le apota tarjeta inteligente teniendo mayor control sobre los procesos que lleva a cabos, sino que le permite hacer uso de esta en cualquier lugar, desde el que tenga acceso a Internet, sin tener que cuestionarse la seguridad de la comunicación.

7.1.2 Conclusiones sobre la solución

La solución ideada se basa en gran medida en la investigación realizada por Jeff King y Andres dos Santos [1] y que fue descrita brevente en la sección 2.2.2. En su desarrollo se propone un mecanimos de seguridad para la autenticación de las comunicaciones entre una persona y un dispositivo seguro basado en la generación de un CAPTCHA. Para la generación del CAPTCHA, en lugar de basarse en las transformaciones típicas asociadas a esta tecnología, deciden elaborar una imagen 3D y aprovechar así las deficiencias de los algoritmos de reconocimiento de imágenes en este ámbito. Las transformaciones 3D permiten la inclusión de sombras, efectos de reflejo y planos que añaden gran cantidad de ruido en la composición. Sin embargo, el procesamiento necesario para realizar este tipo composiciones constituye un gran coste de recursos.

En este proyecto se ha pretendido elaborar un mecanismo de autenticación que cumpla las mismas características que el desarrollado por Jeff King y Andres dos Santos pero cuyo coste de generación sea más ligero. Para ello se realizó un amplio estudio sobre las técnicas utilizadas en la generación de CAPTCHAs, encontrando resultados modernos y adaptables a nuestros objetivos, como son los desarrollados por la Universidad de Pensilvania.

Un aspecto que quedaba abierto en la solución descrita por Jeff King y Andres dos Santos era donde se iban a generar los mecanismos de autenticación. En su estudio no llegan a desarrollar un prototipo, tan sólo plantean las bases del protocolo, pero la necesidad de incluir una nueva entidad en el escenario que sea capaz de construir el mecanismo es obvia, al menos hasta que la capacidad de las tarjetas inteligentes evolucione lo suficiente como para almacenar un sistema de bases de datos y realizar procesos no criptográfico de una forma eficiente.

Otro aspecto que supone una mejora del protocolo resultante en este proyecto frente al establecido en el protocolo con CAPTCHAS en 3D es la inclusión de una clave OTP para la verificación de la comunicación. Jeff King y Andres dos Santos proponían que el usuario, sino estaba conforme con el contenido que aparecía en el mecanismo de seguridad simplemente extraerán la tarjeta inteligente del dispositivo lector, por lo tanto, no existía un el acto de confirmación de la operación, sino que se basaba en la no negación. Este método de estaba expuesto a múltiples ataques. Por lo tanto, era necesario incluir un mecanismo de confirmación. En un primer momento se pensó en el PIN con elemento de validación del proceso, pero al haber sido introducido por el usuario para acceder al sistema podía estar comprometido. Lo mismo sucedía si se acordaba una clave fija de confirmación, tras el primer uso podría estar comprometida. Este razonamiento llevo a pensar en la utilización de una OTP como

confirmación de operación, para que la tarjeta inteligente pudiera validarla tenía que conocerla de antemano y relacionarla con un único mensaje, por esta razón, y dadas las capacidades criptográficas de la smart card, se creyón conveniente que fuera está la que la generará y se la comunicará al usuario incluyéndola en el mecanismo autenticado, así el usuario no tiene ninguna de duda sobre el origen de la clave.

7.1.3 Conocimiento adquiridos en la carrera aplicados en el proyecto

El trabajo desarrollado a lo largo de este proyecto incluye técnicas de muchas de las áreas relacionadas con las ciencias informáticas.

En un principio el ámbito de la propuesta se realiza en la rama de la **seguridad en las tecnologías de la información** y el trabajo realizado tiene un claro enfoque hacia esta área, pero para su desarrollo ha sido necesario aplicar conocimientos de asociados a campos de estudios.

La **inteligencia artificial**, en especial los problemas complejos de la IA, han tenido gran relevancia en el diseño del protocolo y se ha profundizado mucho en el estado del arte de los algoritmos y técnicas de generación de estos problemas y como se pueden aplicar en el área de la seguridad.

Otro aspecto importante, relacionado con el campo de la **ingeniería software**, ha sido la recogida de requisitos que a ayudado en gran medida a concretar el alcance del problema y definir de forma clara los objetivos del proyecto, que antes de aplicar estas técnicas eran algo abstractos y confusos.

Sin lugar a dudas, en el diseño del mecanismo seguridad para la autenticación de mensajes se han tenido en cuenta, en todo momento, la **usabilidad** de la solución. La elección de los parámetros de transformación de las imágenes ha sido un arduo trabajo.

A nivel de implementación, la solución tomada pasa por un **sistema distribuido** que se encuentra ubicado en ente tres componentes de características muy distintas. El ensambla miento de los procesos que se llevan a cabo en cada uno de los componentes es de vital importancia para el funcionamiento del protocolo, por lo que la aplicación de arquitecturas de tipo cliente servidor ha sido de vital importancia.

Por último, destacar el esfuerzo de investigación realizado a lo largo de este trabajo que me ha permitido inicializarme es este aspecto de la ciencia y que no me había sido posible abordar con anterioridad en mis estudios.

7.2 Futuros desarrollos

Durante la realización de este proyecto se han identificado un conjunto de posibles trabajos que podrían completarlo y que inicialmente no estaban incluidos dentro del alcance y objetivos del mismo. Se indican por tanto a continuación como líneas futuras.

Implementación de más funcionalidades

En este desarrollado se ha acotado en gran medida la implementación del protocolo. El prototipo solo muestra la aplicación de la solución para realizar funciones de firma digital, sin embargo, las tarjetas inteligentes permiten realizar otras funciones criptográficas, como puede se el cifrado de entre otras. Una vez se ha establecido el protocolo de comunicación, no supone un gran coste la ampliación del prototipo permitiendo la realización de otras operaciones criptográficas con la tarjeta inteligente.

Aplicación del mecanismo en otros escenarios

Las comunicaciones con un dispositivo de seguridad no son las únicas que se ven afectadas al ataque por entidad externas maliciosas. En muchos sistemas se realizan comunicaciones en los que el receptor es un usuario y este se ve incapaz de autenticar la autoría del mensaje.

Un ejemplo destacable de esta situación son los ataques mediante *phising* en los que se suplanta identidad de una entidad, como puede ser un banco o un comercio, y se insta al usuario a revelar información personal relevante para hacer un uso maliciosa de esta. Estas comunicaciones se realizan normalmente mediante e-mail, si los mensajes enviados por estas entidades, en lugar de ser simple texto, se construyeran mediante el mecanismo de seguridad diseñado en este trabajo, el usuario podría distinguir lo mensajes de su entidad de los enviado por atacantes.

Otra posible aplicación se encuentra en el comercio electrónico, donde las comunicaciones a través de la red pueden verse atacada con seguridad. Para evitar este tipo de vulnerabilidades se han diseñado protocolos como 3-D Secure que incluyen una clave establecida entre el banco y el usuario. El comprador se identificará ante su banco como propietario de la cuenta sobre la que se realizan los cargos de la compra. Sin embargo no se aplica ningún mecanismo que demuestre que realmente es el banco del usuario quien está pidiendo esta identificación, la inclusión del mecanismo de seguridad, diseñado en este proyecto, en la petición de identificación del usuario hará la comunicación mucho más segura ya que ambas podrían autenticarse.

Referencias

[1] King, J., and dos Santos, Andre.: 'A user-friendly approach to human authentication of message ', Springer-Verlag, LNCS 3570, pp. 225-239, 2005.
[2] Spear, M.: 'Rénxìng: Human Cryptography And Identification A Survey ', Technical report, 2007.
[3] von Ahn, L., Blum, M., Hopper, N.J., Langford, J.: 'CAPTCHA: Using hard AI problems for security '. In: Advances in Cryptology – Eurocrypt 2003. Volume 2656 of Lecture Notes in Computer Science., Springer-Verlag, 2003.
[4] Hopper, N.J., Blum, M.: Secure human identification protocols. In: Advances in Cryptology – Asiacrypt 2001. Volume 2248 of Lecture Notes in Computer Science., Springer-Verlag (2001) 52-66
[5] Naor, M., and Shamir, A.: 'Visual cryptography', Technical report, Jerusalem, Israel, Israel, 1994.
[6] von Ahn, L., Blum, M., and Langford, J., 'Telling Humans and Computers Apart (Automatically) or How Lazy Cryptographers do AI', Vol. 47, No. 2 Communications of the ACM, February 2004.
[7] Coates, A.L., Baird, H.S., and Fateman, R.J.: 'Pessimal print: A Reverse Turing Test', *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR '01)*, Seattle, WA, p.p 1154–1159, 2001.
[8] Blum, M., and Baird, H.S., *First Workshop on Human Interactive Proofs*, Palo Alto, CA, 2002.
[9] Chellapilla, K., Larson, K., Simard, P.Y., and Czerwinski, M.: 'Building Segmentation Based Human-Friendly Human Interactive Proofs (HIPs)', H. S. Baird & D. P. Lopresti (Eds), *Proc., 2nd Int'l Workshop on Human Interactive Proofs*, LNCS Vol. No. 3517, Springer (Berlin), pp. 1–26, May 2005.
[10] Chew, M., and Baird, H.S.,: 'BaffleText: a Human Interactive Proof' Proc., 10th SPIE/IS&T Document Recognition and Retrieval Conf. (DRR2003), Santa Clara, CA, January 23-24, 2003.
[11] Simard, P.Y., Steinkraus, D., Platt, J.: 'Best Practice for Convolutional Neural Networks Applied to Visual Document Analysis', *International Conference on Document Analysis and Recognition (ICDAR),* IEEE Computer Society, Los Alamitos, pp. 958-962, 2003.
[12] K. Chellapilla and P. Y. Simard, "Using Machine Learning to Break Visual Human Interaction Proofs (HIPs)" 2004.
[13] Baird, H.S., and Riopka, T.: 'ScatterType: a Reading CAPTCHA Resistant to Segmentation Attack' Proc., IS&T/SPIE Document Recognition and Retrieval Conf, San Jose, CA, January 16-20, 2005.
[14] H. S. Baird and J. L. Bentley, "Implicit CAPTCHAs" 2005.
[15] M. Blum, L. A. von Ahn, and J. Langford, The CAPTCHA Project, 'Completely Automatic Public Turing Test to tell Computers and Humans Apart," www.captcha.net, Dept. of Computer Science, Carnegie-Mellon Univ. and personal communications, November, 2000.

[16] R. Datta, J. Li, and J. Z. Wang, 'IMAGINATION: A Robust Image-based CAPTCHA Generation System', 2005.
[17] S. Panhuber, M. Haudum, "State of the Art in Anti-Automation Technologies in Today's Web" 2006
[18] T. Giang, F. Liu 'Mouse Intervention CAPTCHA' 2006
[19] R. Datta, J. Li, and J. Z. Wang, "Exploting the Human-Machine Gap in Image Recognitation for Designing CAPTCHAs', 2009.
[20] Jonathan Wilkins Strong CAPTCHA Guidelines Dieciembre 2009
[21] N. Nagarajayya and J. S. Mayer, 'Improving Java Application Performance and Scalability by Reducing Garbage Collection Times and Sizing Memory Using JDK 1.4.1', http://developers.sun.com/mobility/midp/articles/garbagecollection2/index.html.
[22] G. McCluskey, "Thirty ways to improve the performance of your Java programs", 1999.
[23] Java Card™ Platform, Version 2.2.2, Sun Microsystems, Inc., Santa Clara, California 95054, U.S.A, 2006.
[24] http://java.sun.com/products/javacard/dev_kit.html
[25] Sm@rtCafé® Professional Toolkit 2.0, Giesecke & Devrient GmbH., Munich, Germany, 2006

Anexo A

PRESUSPUESTO

En este anexo se detalla el presupuesto requerido para llevar a cabo este proyecto. Se va a separar los gastos según la naturaleza de los recursos, y finalmente se contabilizará el coste total del proyecto.

El proyecto se ha llevado a cabo entre los meses de Noviembre del 2009 y Mayo del 2010, con lo que la duración total del proyecto ha sido de 6 meses. Durante este periodo de tiempo se han empleado una media de 4 horas diarias al desarrollo del proyecto, con lo que contando con 20 días laborables por mes dan un total de 480 horas de trabajo. A continuación se muestran cómo han sido distribuidas esas horas en las diferentes actividades de que forma parte este proyecto.

A.1 Desglose por actividad

En la **Tabla** 108, se hace un desglose con las horas que han sido necesarias para llevar a cabo las distintas actividades del proyecto.

Fases de desarrollo	Horas
F1. Análisis del problema	70
F2. Recogida de requisitos	70

F3. Diseño de solución	80
F4. Implementación prototipo	100
F5. Pruebas	40
F6. Documentación	120
TOTAL	480

Tabla 108. Desglose presupuesto por actividad

A.2 Salarios

Para realizar este proyecto se requiere personal formado, que debe tomar un rol u otro dependiendo de la actividad de las mostradas anteriormente realice. La Tabla 109 muestra el coste por cada hora de trabajo de una persona desempeñando cada uno de los roles que se han tenido en cuenta.

En la Tabla 109 se tiene un sueldo bruto anual y el coste por hora de cada uno de los roles. Para la realización de los cálculos se ha tomado 8 horas como la jornada laboral, 20 días laborables al mes y 12 mensualidades.

Cargo	Coste/Horas	Sueldo Bruto
Analista	20 €	38.400 €/año
Ingeniero de requisitos	20 €	38.400 €/año
Responsable de diseño	20 €	38.400 €/año
Desarrollador	15 €	28.400 €/año
Ingeniero de prueba	20 €	38.400 €/año
Responsable de documentación	15 €	28.400 €/año

Tabla 109. Salarios roles implicados en el desarrollo

A.3 Gasto personal

En la Tabla 110 se muestra el coste total de cada uno de los roles teniendo en cuenta las horas que se han dedicado a cada tarea tal y el coste por hora de cada rol. Tal y como se indica en la tabla, el gasto total en personal será de **Ocho mil quinientos** euros.

Cargo	Horas	Coste/Horas	Total
Analista	70	20 €	1.400 €

Ingeniero de requisitos	70	20 €	1.400 €
Responsable de diseño	80	20 €	1.600 €
Desarrollador	100	15 €	1.500 €
Ingeniero de prueba	40	20 €	800 €
Responsable de documentación	120	15 €	1.800 €
TOTAL	480	20 €	8.500 €

Tabla 110. Desglose gastos en personal

A.4 Recursos de materiales

La Tabla 111 contiene el coste de los recursos materiales utilizados durante el desarrollo de este proyecto. El gasto total de los recursos materiales es de **Tres mil setecientos cuarenta y nueve** euros.

Recurso	Cantidad	Coste
Ordenador personal	1	3.000 €
Microsoft Windows XP Professional	1	280 €
Java Card™ Platform, Version 2.2.2, Sun Microsystems, Inc	1	0 €
Sm@rtCafé® Professional Toolkit 2.0	1	450 €
Lector de tarjeta LTC31 USEB, C3PO, S.A	1	19 €
TOTAL		3.749 €

Tabla 111. Gastos en material

A.5 Gastos indirectos

Se van a incluir unos gastos indirectos del 10% de del total de los costes. En la Tabla 112 se introducen estos gastos para calcular el total de gastos del proyecto. El total de los gastos del proyecto será de **Trece mil cuatrocientos setenta y cuatro** euros.

Recurso	Cantidad	Coste
Gasto personal	1	8.500 €
Gasto de recursos materiales	1	3.749 €

Gastos indirectos	1	1.225 €
TOTAL	1	13.474 €

Tabla 112. Desglose gastos indirectos

A.6 Coste total

A los gastos totales de **Trece mil cuatrocientos setenta y cuatro** euros habrá que añadirles un margen de riesgo del 20% y sobre todo ello un beneficio del 15%. Esto supone un coste total del proyecto de **Dieciocho mil quiniestos noventa y cuatro** euros (18.594 Euros).

Anexo B

DEFINICIONES BÁSICAS DE SEGURIDAD

Protocolo de identificación

Un protocolo de identificación es aquel proceso que permite a una entidad identificarse frente a otra como el propietario legítimo de una clave. Este tipo de protocolos se puede aplicar a una amplia variedad de entornos, por ejemplo: cuando una persona desea diferenciarse como individuo del resto de personas frente a una máquina; cuando un sistema desea identificar personas frente a máquina; cuando se desea diferenciar a personas adultas de menores de edad, etc.

Un **protocolo de identificación simétrico** es aquel en el que las dos entidades implicadas en el proceso de identificación, es decir, tanto la entidad que quiere autenticarse como la entidad verificadora, comparten un secreto que será utilizado como clave.

Sistema de autenticación

Los protocolos de identificación llevan asociados la implementación de un sistema de autenticación. Un sistema de autenticación puede definirse como una tupla de procesos (P, V), donde, P es el proceso que debe llevar a cabo el elemento que desea autenticarse y V es el proceso que debe realizar el elemento ante el que se autentica. Hay que tener en cuenta que dependiendo de la naturaleza del elemento a autenticarse el proceso P tendrá distintas implicaciones. La autenticación de una persona a menudo consiste en verificar su identidad, sin embargo, la autenticación de un objeto significa la confirmación de su procedencia.

A esta definición se le debe añadir una premisa para hacerla completa. Todo sistema de autenticación en *casi todas* las ocasiones debe aceptar una entrada válida, sin embargo, en *toda* ocasión debe desestimar una entrada inválida.

En ningún punto de la definición del sistema de autenticación se indica que tipo de mecanismo debe ser utilizado para realizar su cometido. Existen múltiples y variadas técnicas aplicadas en estos sistemas. Habitualmente en el ámbito de la seguridad los métodos de autenticación se han clasificado según el tipo de clave a verificar. Se pueden encontrar tres categorías:

- **Sistemas basados en algo conocido.** Por ejemplo, una contraseña como al utilizada para acceder al correo electrónico.
- **Sistemas basados en algo poseído.** Como puede ser una tarjeta de identidad o un carnet para poder sacar libros de la biblioteca, o una tarjeta inteligente que tenga almacenado un certificado digital de identificación como es el DNI electrónico, u otro tipo de dispositivo criptográfico.
- **Sistemas basados en una característica física** del usuario o un acto involuntario del mismo como son las verificaciones de huellas dactilares, de voz o de escritura.

Protocolo reto-respuesta

Protocolo de identificación (P, V) donde el proceso V cuando recibe la notificación de que un nuevo elemento se ha incorporado al sistema le envía un reto, el nuevo elemento mediante al proceso P para autenticarse deberá devolverle una respuesta correcta para ese reto.

One Time Password (OTP)

Se define One time Password o clave de un solo uso como un esquema de autenticación donde la clave es utilizada una única vez. Este mecanismo de autenticación es muy eficaz en escenarios en los que la comunicación puedes ser fácilmente vulnerada y la clave pueda ser robada con intención de realizar futuros ataques de suplantación. Sin embargo tiene un alto coste en recursos, ya que es necesario un amplio rango de claves. Como método de autenticación de persona es necesario proporcionarla un dispositivo seguro capaz de memorizar las claves.

Este esquema ha sido muy utilizado por entidades bancarias como autenticación de usuarios en su acceso on-line. Siendo habitual proporcionar una matriz de claves impresas en un papel, hecho que debilita en gran medida la seguridad del mecanismo.

Test de Turing invertido (Reverse Turing Test)

En 1950, Alan Turing ideó un método que permitía evaluar la capacidad de las máquinas para simular la inteligencia humana. El test se lleva a cabo por una persona denominada juez y consiste en enfrentar a una persona y una máquina. El juez realiza una serie de preguntas o retos a cada uno de *concursantes* y basándose en sus respuestas asigna a un participante la identidad de persona y al otro la de máquina. Si el juez falla en su valoración de las identidades implicaría que una máquina es capaz de simular el comportamiento de una persona para los retos que componen el test, es decir, la máquina tiene inteligencia.

Muchos son los investigadores que desde los años 50 han ideado múltiples variaciones de este método para su uso en distintos escenarios. Una de las aplicaciones más utilizadas es el llamado test de Turing invertido o **Protocolo de Identificación Humana (HIP)**. Es test aprovecha los retos realizados en test de Turing para conocer aquellos aspectos en los que la tecnología no ha evolucionado lo suficiente como para suplantar a las persona y los utiliza como método para

diferenciación entre humanos y máquinas. Los CAPTCHAs (Completely Automatic Public Turing test to tell Computers and Humans Apart) es una de las implementaciones de estos test de Turing invertido, en la sección 2.3 se aportará más información sobre este mecanismo.